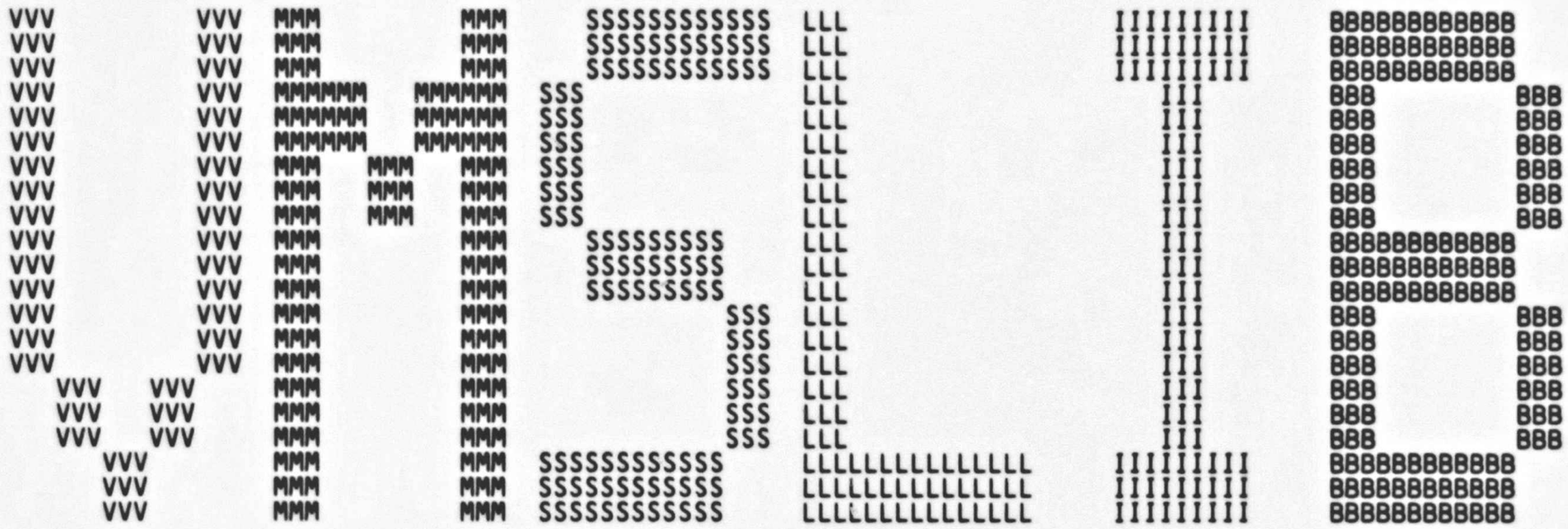
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSSSS LLL IIIIIIIII BBBBBBBBBBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMMMMM MMMMMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSSSSSSSS LLL III BBBBBBBBBBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV VVV MMM MMM SSS LLL III BBB BBB
VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB
VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB
VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB
VVV MMM MMM SSSSSSSSSSS LLLLLLLLLLLLLLL IIIIIIIII BBBBBBBBBBB

FILEID**SCRMISC

```
  SSSSSSSS    CCCCCCCC  RRRRRRRR    MM      MM   IIIIII     SSSSSSSS    CCCCCCCC
  SSSSSSSS    CCCCCCCC  RRRRRRRR    MM      MM   IIIIII     SSSSSSSS    CCCCCCCC
SS          CC          RR      RR  MMMM  MMMM     II     SS          CC
SS          CC          RR      RR  MMMM  MMMM     II     SS          CC
SS          CC          RR      RR  MM  MM  MM     II     SS          CC
SS          CC          RR      RR  MM  MM  MM     II     SS          CC
  SSSSSS    CC          RRRRRRRR    MM      MM     II       SSSSSS    CC
  SSSSSS    CC          RRRRRRRR    MM      MM     II       SSSSSS    CC
        SS  CC          RR  RR      MM      MM     II             SS  CC
        SS  CC          RR  RR      MM      MM     II             SS  CC
        SS  CC          RR    RR    MM      MM     II             SS  CC
        SS  CC          RR    RR    MM      MM     II             SS  CC
SSSSSSSS      CCCCCCCC  RR      RR  MM      MM   IIIIII   SSSSSSSS      CCCCCCCC
SSSSSSSS      CCCCCCCC  RR      RR  MM      MM   IIIIII   SSSSSSSS      CCCCCCCC

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;       1     0001  0 %TITLE 'SCR$MISC - Misc. routines for the screen package'
;       2     0002  0 MODULE SCR$MISC (
;       3     0003  0                 IDENT = 'V04-000'       ! File: SCRMISC.B32 Edit: PLL1005
;       4     0004  0                 ) =
;       5     0005  1 BEGIN
;       6     0006  1 !
;       7     0007  1 !**************************************************************************
;       8     0008  1 !*                                                                        *
;       9     0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
;      10     0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
;      11     0011  1 !*  ALL RIGHTS RESERVED.                                                   *
;      12     0012  1 !*                                                                        *
;      13     0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;      14     0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;      15     0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;      16     0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;      17     0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;      18     0018  1 !*  TRANSFERRED.                                                           *
;      19     0019  1 !*                                                                        *
;      20     0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;      21     0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;      22     0022  1 !*  CORPORATION.                                                           *
;      23     0023  1 !*                                                                        *
;      24     0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;      25     0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
;      26     0026  1 !*                                                                        *
;      27     0027  1 !*                                                                        *
;      28     0028  1 !**************************************************************************
;      29     0029  1 !
;      30     0030  1
;      31     0031  1 !++
;      32     0032  1 ! FACILITY:     Screen Management
;      33     0033  1 !
;      34     0034  1 ! ABSTRACT:
;      35     0035  1 !
;      36     0036  1 !       This module contains routines which can perform their functions
;      37     0037  1 !       independently of the other screen routines.  These include
;      38     0038  1 !       information reporting and initialization routines.
;      39     0039  1 !
;      40     0040  1 ! ENVIRONMENT:  User mode, Shared library routines.
;      41     0041  1 !
;      42     0042  1 ! AUTHOR: P. Levesque, CREATION DATE: 18-Oct-1982
;      43     0043  1 !
;      44     0044  1 ! MODIFIED BY:
;      45     0045  1 !
;      46     0046  1 ! 1-001 - Original.  PLL 18-Oct-1982
;      47     0047  1 ! 1-002 - Use VMS logicals to point to require files.  PLL 24-Jan-1983
;      48     0048  1 ! 1-003 - GET_CHAR should move a byte, not word, to pagesize.
;      49     0049  1 !         PLL 31-Jan-1983
;      50     0050  1 ! 1-004 - In GET_CHAR, if the terminal is not a vt52 or vt100, skip
;      51     0051  1 !         call to SCR$PUT_SCREEN to output mode setting.  PLL 19-Jul-1983
;      52     0052  1 ! 1-005 - A fix to the parameter checking in LIB$SCREEN_INFO so that omitted
;      53     0053  1 !         parameters are handled correctly.  PLL 21-Aug-1984
;      54     0054  1 !--
;      55     0055  1
```

```
;    57          0056  1 %SBTTL 'Declarations'
;    58          0057  1 !
;    59          0058  1 ! SWITCHES:
;    60          0059  1 !
;    61          0060  1
;    62          0061  1 !
;    63          0062  1 ! LINKAGES:
;    64          0063  1 !
;    65          0064  1 !       NONE
;    66          0065  1 !
;    67          0066  1 ! TABLE OF CONTENTS:
;    68          0067  1 !
;    69          0068  1
;    70          0069  1 FORWARD ROUTINE
;    71          0070  1
;    72          0071  1 !    The LIB$ entry points
;    73          0072  1
;    74          0073  1         LIB$SCREEN_INFO,                ! Screen information retrieval
;    75          0074  1         LIB$SET_OUTPUT,                 ! Establish terminal for output
;    76          0075  1
;    77          0076  1 !    The SCR$ entry points
;    78          0077  1
;    79          0078  1         SCR$SCREEN_INFO,                ! Screen information retrieval
;    80          0079  1         SCR$SET_OUTPUT,                 ! Establish terminal for output
;    81          0080  1         SCR$STOP_OUTPUT,                ! Stop output to terminal or screen buffer
;    82          0081  1
;    83          0082  1 !    Local subroutines
;    84          0083  1         CREATE,                 ! Create an RMS file
;    85          0084  1         EXIT_HANDLER,           ! Image exit handler
;    86          0085  1         GET_CHAR;               ! Get terminal characteristics
;    87          0086  1
;    88          0087  1
;    89          0088  1 ! The following is equated via a GLOBAL BIND
;    90          0089  1 !       LIB$STOP_OUTPUT = SCR$STOP_OUTPUT
;    91          0090  1
;    92          0091  1 !
;    93          0092  1 ! INCLUDE FILES
;    94          0093  1 !
;    95          0094  1
;    96          0095  1 REQUIRE 'SRC$:SCRPROLOG';               ! defines psects, macros, tcb,
;    97          0170  1                                         !  wcb, & terminal symbols
;    98          0171  1 REQUIRE 'LIB$:STRLNK';                  ! Linkage to LIB$ANALYZE_SDESC_R2
;    99          0356  1 OWN
;   100          0357  1     SCR$EXITBLOCK   : VECTOR [4] INITIAL (0, 0, 1, 0),
;   101          0358  1                                         ! 1st longword is ?
;   102          0359  1                                         ! 2nd longword is exit handler
;   103          0360  1                                         ! 3rd longword is ?
;   104          0361  1                                         ! 4th longword is exit status
;   105          0362  1     SCR$L_EXITSTS ;                     ! filled in before exit handler
;   106          0363  1                                         ! called
;   107          0364  1
;   108          0365  1 !+
;   109          0366  1 ! $GETDVI storage
;   110          0367  1 ! ---------------
;   111          0368  1 !-
;   112          0369  1 OWN
;   113          0370  1     SCR$AB_DEVCLASS,
```

```
;  114          0371  1      SCR$AB_DEVTYPE,
;  115          0372  1      SCR$AW_DEVBUFSIZ,
;  116          0373  1      SCR$AL_DEVDEPEND;
;  117          0374  1  GLOBAL
;  118          0375  1      SCR$AL_DEVDEPND2 : BLOCK [4, BYTE] ;
;  119          0376  1
;  120          0377  1  OWN
;  121          0378  1      SCR$A_ITMLST : VECTOR [5*3 + 1] INITIAL (
;  122          0379  1                              DVI$_DEVCLASS   ^16 + 4, 0, 0,
;  123          0380  1                              DVI$_DEVTYPE    ^16 + 4, 0, 0,
;  124          0381  1                              DVI$_DEVBUFSIZ  ^16 + 4, 0, 0,
;  125          0382  1                              DVI$_DEVDEPEND  ^16 + 4, 0, 0,
;  126          0383  1                              DVI$_DEVDEPEND2 ^16 + 4, 0, 0,
;  127          0384  1                              0 ) ;
;  128          0385  1  BIND
;  129          0386  1      SCR$_INFOCLASS = SCR$A_ITMLST + 4,
;  130          0387  1      SCR$_INFOTYPE  = SCR$A_ITMLST + 16,
;  131          0388  1      SCR$_INFOSIZ   = SCR$A_ITMLST + 28,
;  132          0389  1      SCR$_INFODEP   = SCR$A_ITMLST + 40,
;  133          0390  1      SCR$_INFODEP2  = SCR$A_ITMLST + 52 ;
;  134          0391  1
;  135          0392  1  !
;  136          0393  1  ! EXTERNAL REFERENCES
;  137          0394  1  !
;  138          0395  1
;  139          0396  1  EXTERNAL ROUTINE
;  140          0397  1      SCR$$GET_TYPE_R3 : GET_TYPE_LINK, ! Get device type
;  141          0398  1      LIB$ANALYZE_SDESC_R2 : LIB$ANALYZE_SDESC_JSB_LINK,
;  142          0399  1      LIB$ASSIGN,                   ! Assign a channel
;  143          0400  1      LIB$FREE_EF,                  ! Free a local event flag number
;  144          0401  1      LIB$FREE_VM,                  ! Heap storage deallocator
;  145          0402  1      LIB$GET_EF,                   ! Get a local event flag number
;  146          0403  1      LIB$GET_VM,                   ! Heap storage allocator
;  147          0404  1      LIB$LP_LINES,                 ! Default lines per page
;  148          0405  1      SCR$PUT_SCREEN,               ! Put text to screen
;  149          0406  1      SCR$SET_SCROLL;               ! Set a scrolling region
;  150          0407  1
;  151          0408  1  EXTERNAL
;  152          0409  1      SCR$L_CUROUTPUT : REF BLOCK [, BYTE], ! Pointer to current TCB
;  153          0410  1      SCR$L_FLINKHEAD;              ! Head of chain of TCB's
;  154          0411  1  !<BLF/PAGE>
```

```
;  156     0412  1 %SBTTL 'LIB$SCREEN_INFO - Screen Information Retrieval'
;  157     0413  1 GLOBAL ROUTINE LIB$SCREEN_INFO (
;  158     0414  1                                 FLAGS           : REF VECTOR [,LONG],
;  159     0415  1                                 DEV_TYPE        : REF VECTOR [,BYTE],
;  160     0416  1                                 LINE_WIDTH      : REF VECTOR [,WORD],
;  161     0417  1                                 LINES_PER_PAGE  : REF VECTOR [,WORD]
;  162     0418  1                                 ) =
;  163     0419  1 !++
;  164     0420  1 ! FUNCTIONAL DESCRIPTION:
;  165     0421  1 !
;  166     0422  1 !       This routine
;  167     0423  1 !
;  168     0424  1 ! CALLING SEQUENCE:
;  169     0425  1 !
;  170     0426  1 !       ret_status.wlc.v = LIB$SCREEN_INFO (FLAGS.wl.r
;  171     0427  1 !                                           [,DEV_TYPE.wb.r
;  172     0428  1 !                                           [,LINE_WIDTH.ww.r
;  173     0429  1 !                                           [,LINES_PER_PAGE.ww.r]]])
;  174     0430  1 !
;  175     0431  1 ! FORMAL PARAMETERS:
;  176     0432  1 !
;  177     0433  1 !       FLAGS.wl.r      Rendition code for current window.
;  178     0434  1 !                       Values:
;  179     0435  1 !                           SCR$M_BLINK         display characters blinking.
;  180     0436  1 !                           SCR$M_BOLD          display characters in
;  181     0437  1 !                                               higher-than-normal intensity.
;  182     0438  1 !                           SCR$M_NORMAL        display characters using
;  183     0439  1 !                                               rendition associated with
;  184     0440  1 !                                               window.
;  185     0441  1 !                           SCR$M_REVERSE       display characters in reverse
;  186     0442  1 !                                               video -- i.e., using opposite
;  187     0443  1 !                                               rendition from window default.
;  188     0444  1 !                           SCR$M_UNDERLINE     display characters underlined.
;  189     0445  1 !
;  190     0446  1 !       DEV_TYPE.wb.r           Optional.
;  191     0447  1 !
;  192     0448  1 !       LINE_WIDTH.ww.r         Optional.
;  193     0449  1 !
;  194     0450  1 !       LINES_PER_PAGE.ww.r     Optional.
;  195     0451  1 !
;  196     0452  1 ! IMPLICIT INPUTS:
;  197     0453  1 !
;  198     0454  1 !       NONE
;  199     0455  1 !
;  200     0456  1 ! IMPLICIT OUTPUTS:
;  201     0457  1 !
;  202     0458  1 !       NONE
;  203     0459  1 !
;  204     0460  1 ! COMPLETION STATUS:
;  205     0461  1 !
;  206     0462  1 !       SS$_NORMAL      Normal successful completion
;  207     0463  1 !
;  208     0464  1 ! SIDE EFFECTS:
;  209     0465  1 !
;  210     0466  1 !       NONE
;  211     0467  1 !--
;  212     0468  1
```

```
2        BEGIN
2
2        BUILTIN
2            ACTUALCOUNT,
2            ACTUALPARAMETER ;
2
2        LOCAL
2            LOC_BUFFER : BLOCK [SCR$K_LENGTH, BYTE] ;          ! Local buffer
2
2   !+
2   ! Get terminal information into our local buffer.
2   !-
2        SCR$SCREEN_INFO ( LOC_BUFFER ) ;
2
2   !+
2   ! If no args, just return.
2   !-
2        IF ACTUALCOUNT() EQL 0
2        THEN
2            RETURN ( SS$_NORMAL ) ;
2
2   !+
2   ! If FLAGS argument present, return the FLAGS value.
2   !-
2        IF ACTUALCOUNT() GEQ 1
2        THEN
3            BEGIN
3            IF ACTUALPARAMETER(1) NEQ 0
3            THEN
3                FLAGS[0] = .LOC_BUFFER [SCR$L_FLAGS]
3            END
2        ELSE
2            RETURN ( SS$_NORMAL ) ;
2
2   !+
2   ! If DEV_TYPE argument present, return the DEV_TYPE value.
2   !-
2        IF ACTUALCOUNT() GEQ 2
2        THEN
3            BEGIN
3            IF ACTUALPARAMETER(2) NEQ 0
3            THEN
3                DEV_TYPE[0] = .LOC_BUFFER [SCR$B_DEVTYPE]
3            END
2        ELSE
2            RETURN ( SS$_NORMAL ) ;
2
2   !+
2   ! If LINE_WIDTH argument present, return the LINE_WIDTH value.
2   !-
2        IF ACTUALCOUNT() GEQ 3
2        THEN
3            BEGIN
3            IF ACTUALPARAMETER(3) NEQ 0
3            THEN
3                LINE_WIDTH[0] = .LOC_BUFFER [SCR$W_WIDTH]
3            END
```

```
;  270          0526  2     ELSE
;  271          0527  2         RETURN ( SS$_NORMAL ) ;
;  272          0528  2
;  273          0529  2 !+
;  274          0530  2 ! If LINES_PER_PAGE argument present, return the LINES_PER_PAGE value.
;  275          0531  2 !-
;  276          0532  2     IF ACTUALCOUNT() GEQ 4
;  277          0533  2     THEN
;  278          0534  2         IF ACTUALPARAMETER(4) NEQ 0
;  279          0535  2         THEN
;  280          0536  2             LINES_PER_PAGE[0] = .LOC_BUFFER [SCR$W_PAGESIZE] ;
;  281          0537  2
;  282          0538  2     RETURN ( SS$_NORMAL ) ;
;  283          0539  2
;  284          0540  1     END;                                    ! End of routine LIB$SCREEN_INFO


;                                                                .TITLE  SCR$MISC SCR$MISC - Misc. routines for the scre
                                                                                 en packag
                                                                .IDENT  \V04-000\

                                                                .PSECT  _LIB$DATA,NOEXE,  PIC,2

                00000000  00000001  00000000  00000000  00000 SCR$EXITBLOCK:
                                                                .LONG   0, 0, 1, 0                                      ;
                                                            00010 SCR$L_EXITSTS:
                                                                .BLKB   4
                                                            00014 SCR$AB_DEVCLASS:
                                                                .BLKB   4
                                                            00018 SCR$AB_DEVTYPE:
                                                                .BLKB   4
                                                            0001C SCR$AW_DEVBUFSIZ:
                                                                .BLKB   4
                                                            00020 SCR$AL_DEVDEPEND:
                                                                .BLKB   4
                                                            00024 SCR$AL_DEVDEPND2::
                                                                .BLKB   4
00000000  00000000  00060004  00000000  00000000  00040004  00028 SCR$A_ITMLST:
                                                                .LONG   262148, 0, 0, 393220, 0, 0, 524292, 0, 0, -     ;
00000000  00000000  000A0004  00000000  00000000  00080004  00040                   655364, 0, 0, 1835012, 0, 0, 0      ;
                    00000000  00000000  00000000  001C0004  00058                                                       ;


                                                        SCR$_INFOCLASS=     SCR$A_ITMLST+4
                                                        SCR$_INFOTYPE=      SCR$A_ITMLST+16
                                                        SCR$_INFOSIZ=       SCR$A_ITMLST+28
                                                        SCR$_INFODEP=       SCR$A_ITMLST+40
                                                        SCR$_INFODEP2=      SCR$A_ITMLST+52
                                                                .EXTRN  SCR$$GET_TYPE_R3
                                                                .EXTRN  LIB$ANALYZE_SDESC_R2
                                                                .EXTRN  LIB$ASSIGN, LIB$FREE_EF
                                                                .EXTRN  LIB$FREE_VM, LIB$GET_EF
                                                                .EXTRN  LIB$GET_VM, LIB$LP_LINES
                                                                .EXTRN  SCR$PUT_SCREEN, SCR$SET_SCROLL
                                                                .EXTRN  SCR$L_COROUTPUT
                                                                .EXTRN  SCR$L_FLINKHEAD

                                                                .PSECT  _LIB$CODE,NOWRT,  SHR,  PIC,2
```

```
                                 0000 00000        .ENTRY  LIB$SCREEN_INFO, Save nothing                  ; 0413
               5E             14   C2 00002        SUBL2   #20, SP
                              5E   DD 00005        PUSHL   SP                                             ; 0481
        0000V  CF             01   FB 00007        CALLS   #1, SCR$SCREEN_INFO
                              6C   95 0000C        TSTB    (AP)                                           ; 0486
                              36   13 0000E        BEQL    4$
                        04    AC   D5 00010        TSTL    4(AP)                                          ; 0496
                              04   13 00013        BEQL    1$
           04  BC             6E   D0 00015        MOVL    LOC_BUFFER, @FLAGS                             ; 0498
               02             6C   91 00019 1$:    CMPB    (AP), #2                                       ; 0506
                              28   1F 0001C        BLSSU   4$
                        08    AC   D5 0001E        TSTL    8(AP)                                          ; 0509
                              05   13 00021        BEQL    2$
           08  BC           08    AE   90 00023    MOVB    LOC_BUFFER+8, @DEV_TYPE                        ; 0511
               03             6C   91 00028 2$:    CMPB    (AP), #3                                       ; 0519
                              19   1F 0002B        BLSSU   4$
                        0C    AC   D5 0002D        TSTL    12(AP)                                         ; 0522
                              05   13 00030        BEQL    3$
           0C  BC           04    AE   B0 00032    MOVW    LOC_BUFFER+4, @LINE_WIDTH                      ; 0524
               04             6C   91 00037 3$:    CMPB    (AP), #4                                       ; 0532
                              0A   1F 0003A        BLSSU   4$
                        10    AC   D5 0003C        TSTL    16(AP)                                         ; 0534
                              05   13 0003F        BEQL    4$
           10  BC           06    AE   B0 00041    MOVW    LOC_BUFFER+6, @LINES_PER_PAGE                  ; 0536
               50             01   D0 00046 4$:    MOVL    #1, R0                                         ; 0538
                                   04 00049        RET                                                    ; 0540

; Routine Size:  74 bytes,    Routine Base:  _LIB$CODE + 0000

;  285         0541  1 !<BLF/PAGE>
```

```
;   287          0542  1 %SBTTL 'LIB$SET_OUTPUT - Set Terminal or Screen Buffer for Output'
;   288          0543  1 GLOBAL ROUTINE LIB$SET_OUTPUT (
;   289          0544  1                                 CHAN          : REF VECTOR [,WORD],
;   290          0545  1                                 FILE_SPEC     : REF BLOCK [,BYTE],
;   291          0546  1                                 USER_ROUTINE  : REF VECTOR [,LONG],
;   292          0547  1                                 USER_ARG      : REF VECTOR [,LONG],
;   293          0548  1                                 STREAM        : REF VECTOR [,LONG]
;   294          0549  1                               ) =
;   295          0550  1 !++
;   296          0551  1 ! FUNCTIONAL DESCRIPTION:
;   297          0552  1 !
;   298          0553  1 !       This routine sets up a device to receive output.  If this is
;   299          0554  1 !       the first call, obtain device characteristics.
;   300          0555  1 !
;   301          0556  1 !       If this is the first call for the screen than a screen
;   302          0557  1 !       control block is allocated, a channel is assigned, device
;   303          0558  1 !       characteristics are obtained.  If it is an unknown device then
;   304          0559  1 !       the channel is deassigned (no QIO output).  If a file
;   305          0560  1 !       specification is present and no user routine is declared then
;   306          0561  1 !       the file is opened via RMS.
;   307          0562  1 !
;   308          0563  1 !       At output time the user-supplied routine, if present, will be
;   309          0564  1 !       called with the following parameters:
;   310          0565  1 !
;   311          0566  1 !               AP -->  4
;   312          0567  1 !                       Optional user supplied argument.
;   313          0568  1 !                       Address of channel number (0 if none)
;   314          0569  1 !                       Address of string dsc to output
;   315          0570  1 !                       Address of stream number
;   316          0571  1 !
;   317          0572  1 !
;   318          0573  1 ! CALLING SEQUENCE:
;   319          0574  1 !
;   320          0575  1 !       ret_status.wlc.v = LIB$SET_OUTPUT (CHAN.rw.r
;   321          0576  1 !                                          [,FILE_SPEC.rt.dx
;   322          0577  1 !                                          [,USER_ROUTINE.zem.rp
;   323          0578  1 !                                          [,USER_ARG.rl.r
;   324          0579  1 !                                          [,STREAM.wl.r]]]]
;   325          0580  1 !
;   326          0581  1 ! FORMAL PARAMETERS:
;   327          0582  1 !
;   328          0583  1 !       CHAN.rw.r               Address of stream number.
;   329          0584  1 !
;   330          0585  1 !       FILE_SPEC.rt.dx         Optional.  Address of descriptor of
;   331          0586  1 !                               file specification.
;   332          0587  1 !
;   333          0588  1 !       USER_ROUTINE.zem.rp     Optional.  Address of output routine to
;   334          0589  1 !                               call for output.
;   335          0590  1 !
;   336          0591  1 !       USER_ARG.rl.r           Optional.  Address of argument to be
;   337          0592  1 !                               passed to user-specified output routine.
;   338          0593  1 !
;   339          0594  1 !       STREAM.wl.r             Optional.  Address of longword to
;   340          0595  1 !                               receive previous stream.
;   341          0596  1 !
;   342          0597  1 ! IMPLICIT INPUTS:
;   343          0598  1 !
```

```
;   344     0599  1 !       NONE
;   345     0600  1 !
;   346     0601  1 ! IMPLICIT OUTPUTS:
;   347     0602  1 !
;   348     0603  1 !       NONE
;   349     0604  1 !
;   350     0605  1 ! COMPLETION STATUS:
;   351     0606  1 !
;   352     0607  1 !       SS$_NORMAL      Normal successful completion
;   353     0608  1 !
;   354     0609  1 ! SIDE EFFECTS:
;   355     0610  1 !
;   356     0611  1 !       NONE
;   357     0612  1 !--
;   358     0613  1
;   359     0614  2     BEGIN
;   360     0615  2
;   361     0616  2     BUILTIN
;   362     0617  2         ACTUALCOUNT,
;   363     0618  2         ACTUALPARAMETER,
;   364     0619  2         CALLG ;
;   365     0620  2
;   366     0621  2     LOCAL
;   367     0622  2         NEW_CALL_LIST : VECTOR [6] ;    ! More be one longword longer
;   368     0623  2                                         ! than maximum number of
;   369     0624  2                                         ! arguments to this routine.
;   370     0625  2
;   371     0626  2 !+
;   372     0627  2 ! Constuct a new call list which is our original call list including
;   373     0628  2 ! the number of arguments.
;   374     0629  2 !-
;   375     0630  2     DECR I FROM ACTUALCOUNT() TO 0
;   376     0631  2     DO
;   377     0632  3         BEGIN
;   378     0633  3         NEW_CALL_LIST[.I] = ACTUALPARAMETER(.I) ;
;   379     0634  2         END;
;   380     0635  2
;   381     0636  2 !+
;   382     0637  2 ! Promote the CHAN argument to by-value in our new call list.
;   383     0638  2 !-
;   384     0639  2     NEW_CALL_LIST[1] = ..NEW_CALL_LIST[1] ;
;   385     0640  2
;   386     0641  2     RETURN ( CALLG ( NEW_CALL_LIST, SCR$SET_OUTPUT )) ;
;   387     0642  2
;   388     0643  1     END;                                ! End of routine LIB$SET_OUTPUT


                        0000 00000         .ENTRY  LIB$SET_OUTPUT, Save nothing                    ; 0543
              5E      18  C2 00002         SUBL2   #24, SP                                         ;
              50      6C  9A 00005         MOVZBL  (AP), I                                         ; 0630
                      50  D6 00008         INCL    I                                               ;
                      05  11 0000A         BRB     2$                                              ;
            6E40    6C40  D0 0000C 1$:     MOVL    (AP)[I], NEW_CALL_LIST[I]                       ; 0633
              F8      50  F4 00011 2$:     SOBGEQ  I, 1$                                           ; 0630
```

```
                 04  AE        04  BE  D0 00014     MOVL   @NEW_CALL_LIST+4, NEW_CALL_LIST+4        ; 0639
              0000V  CF            6E  FA 00019     CALLG  NEW_CALL_LIST, SCR$SET_OUTPUT            ; 0641
                                       04 0001E     RET                                             ; 0643

; Routine Size:  31 bytes,    Routine Base:  _LIB$CODE + 004A

;  389          0644  1 !<BLF/PAGE>
```

```
;  391          0645  1 %SBTTL 'LIB$STOP_OUTPUT - Stop Output to Terminal or Screen Buffer'
;  392          0646  1 ! GLOBAL ROUTINE LIB$STOP_OUTPUT (
;  393          0647  1 !                                  CHAN : REF VECTOR [,WORD]
;  394          0648  1 !                                 ) =
;  395          0649  1 !++
;  396          0650  1 ! FUNCTIONAL DESCRIPTION:
;  397          0651  1 !
;  398          0652  1 !       This routine deaccesses a stream established for output.
;  399          0653  1 !
;  400          0654  1 ! CALLING SEQUENCE:
;  401          0655  1 !
;  402          0656  1 !       ret_status.wlc.v = LIB$STOP_OUTPUT (CHAN.rw.r)
;  403          0657  1 !
;  404          0658  1 ! FORMAL PARAMETERS:
;  405          0659  1 !
;  406          0660  1 !       CHAN.rw.r       Address of channel number
;  407          0661  1 !
;  408          0662  1 ! IMPLICIT INPUTS:
;  409          0663  1 !
;  410          0664  1 !       NONE
;  411          0665  1 !
;  412          0666  1 ! IMPLICIT OUTPUTS:
;  413          0667  1 !
;  414          0668  1 !       NONE
;  415          0669  1 !
;  416          0670  1 ! COMPLETION STATUS:
;  417          0671  1 !
;  418          0672  1 !       SS$_NORMAL      Normal successful completion
;  419          0673  1 !
;  420          0674  1 ! SIDE EFFECTS:
;  421          0675  1 !
;  422          0676  1 !       The channel or ISI is deassigned.
;  423          0677  1 !--
;  424          0678  1
;  425          0679  1 !+
;  426          0680  1 ! Equate to SCR$ entry point.
;  427          0681  1 !-
;  428          0682  1
;  429          0683  1 GLOBAL BIND ROUTINE LIB$STOP_OUTPUT = SCR$STOP_OUTPUT ;
;  430          0684  1 !<BLF/PAGE>
```

```
;  432     0685  1 %SBTTL 'SCR$SCREEN_INFO - Get Screen Information'
;  433     0686  1 GLOBAL ROUTINE SCR$SCREEN_INFO (
;  434     0687  1                                 CONTROL_BLOCK : REF BLOCK [,BYTE]
;  435     0688  1                                ) =
;  436     0689  1 !++
;  437     0690  1 ! FUNCTIONAL DESCRIPTION:
;  438     0691  1 !
;  439     0692  1 !       This routine obtains information about the current output
;  440     0693  1 !       screen.  It returns this information in the user-specified
;  441     0694  1 !       buffer.
;  442     0695  1 !
;  443     0696  1 ! CALLING SEQUENCE:
;  444     0697  1 !
;  445     0698  1 !       ret_status.wlc.v = SCR$SCREEN_INFO (CONTROL_BLOCK.wab.r)
;  446     0699  1 !
;  447     0700  1 ! FORMAL PARAMETERS:
;  448     0701  1 !
;  449     0702  1 !       CONTROL_BLOCK.wab.r     Address of buffer to receive information
;  450     0703  1 !
;  451     0704  1 ! IMPLICIT INPUTS:
;  452     0705  1 !
;  453     0706  1 !       NONE
;  454     0707  1 !
;  455     0708  1 ! IMPLICIT OUTPUTS:
;  456     0709  1 !
;  457     0710  1 !       NONE
;  458     0711  1 !
;  459     0712  1 ! COMPLETION STATUS:
;  460     0713  1 !
;  461     0714  1 !       SS$_NORMAL      Normal successful completion
;  462     0715  1 !
;  463     0716  1 ! SIDE EFFECTS:
;  464     0717  1 !
;  465     0718  1 !       NONE
;  466     0719  1 !--
;  467     0720  1
;  468     0721  2     BEGIN
;  469     0722  2     LOCAL
;  470     0723  2         DEV_TYPE,                   ! Device type
;  471     0724  2         STATUS,
;  472     0725  2         TCB : REF BLOCK [,BYTE] ;       ! Address of current terminal
;  473     0726  2                                         ! control block
;  474     0727  2
;  475     0728  2
;  476     0729  2     STATUS = SCR$$GET_TYPE_R3 (-1; TCB, DEV_TYPE) ;
;  477     0730  2                                         ! Get current terminal control block
;  478     0731  2     IF NOT .STATUS THEN RETURN (.STATUS);
;  479     0732  2
;  480     0733  2     CONTROL_BLOCK [SCR$L_FLAGS] = 0 ; ! Preset to zero
;  481     0734  2
;  482     0735  2     IF .DEV_TYPE NEQ UNKNOWN
;  483     0736  2     THEN
;  484     0737  3         BEGIN
;  485     0738  3         IF .DEV_TYPE NEQ VTFOREIGN
;  486     0739  3         THEN
;  487     0740  3             CONTROL_BLOCK [SCR$L_FLAGS] =
;  488     0741  3                     .CONTROL_BLOCK [SCR$L_FLAGS] OR SCR$M_SCREEN ;
```

```
0742 2        END;
0743 2
0744 2        CONTROL_BLOCK [SCR$B_DEVTYPE]   = .TCB [SCR$B_DEVTYP] ;
0745 2        CONTROL_BLOCK [SCR$W_WIDTH]     = .TCB [SCR$W_DEVWIDTH] ;
0746 2        CONTROL_BLOCK [SCR$W_PAGESIZE]  = .TCB [SCR$W_DEVPAGSIZ] ;
0747 2
0748 2 !+
0749 2 ! Move bits
0750 2 !-
0751 3        BEGIN          ! Bit movement
0752 3        BIND SOURCE_FIELD = TCB [SCR$L_DEVDEPND2] ;
0753 3        BIND DEST_FIELD   = CONTROL_BLOCK [SCR$L_FLAGS] ;
0754 3
0755 3        MAP SOURCE_FIELD : BLOCK [,BYTE],
0756 3            DEST_FIELD   : BLOCK [,BYTE];
0757 3
0758 3        DEST_FIELD [SCR$V_ANSICRT]   = .SOURCE_FIELD [TT2$V_ANSICRT] ;
0759 3        DEST_FIELD [SCR$V_REGIS]     = .SOURCE_FIELD [TT2$V_REGIS] ;
0760 3        DEST_FIELD [SCR$V_BLOCK]     = .SOURCE_FIELD [TT2$V_BLOCK] ;
0761 3        DEST_FIELD [SCR$V_AVO]       = .SOURCE_FIELD [TT2$V_AVO] ;
0762 3        DEST_FIELD [SCR$V_EDIT]      = .SOURCE_FIELD [TT2$V_EDIT] ;
0763 3        DEST_FIELD [SCR$V_DECCRT]    = .SOURCE_FIELD [TT2$V_DECCRT] ;
0764 2        END ;          ! Bit movement
0765 2
0766 2        RETURN (SS$_NORMAL);
0767 1        END;                                          ! End of routine SCR$SCREEN_INFO


                                   000C 00000        .ENTRY  SCR$SCREEN_INFO, Save R2,R3       ; 0686
                      50        01  CE 00002        MNEGL   #1, R0                            ; 0729
               00000000G        00  16 00005        JSB     SCR$$GET_TYPE_R3                  ;
                      5B        50  E9 0000B        BLBC    STATUS, 2$                        ; 0731
                      50   04   AC  D0 0000E        MOVL    CONTROL_BLOCK, R0                 ; 0733
                                60  D4 00012        CLRL    (R0)                              ;
                                52  D5 00014        TSTL    DEV_TYPE                          ; 0735
                                08  13 00016        BEQL    1$                                ;
                      04        52  D1 00018        CMPL    DEV_TYPE, #4                      ; 0738
                                03  13 0001B        BEQL    1$                                ;
                      60        01  88 0001D        BISB2   #1, (R0)                          ; 0741
                08    A0   0B   A1  90 00020 1$:    MOVB    11(TCB), 8(R0)                    ; 0744
                04    A0   0C   A1  D0 00025        MOVL    12(TCB), 4(R0)                    ; 0745
                      51   44   A1  9E 0002A        MOVAB   68(R1), R1                        ; 0752
  60     01     01         03   A1  F0 0002E        INSV    3(R1), #1, #1, (R0)               ; 0758
  52     61     01              19  EF 00034        EXTZV   #25, #1, (R1), R2                 ; 0759
  60     01     02              52  F0 00039        INSV    R2, #2, #1, (R0)                  ;
  52     61     01              1A  EF 0003E        EXTZV   #26, #1, (R1), R2                 ; 0760
  60     01     03              52  F0 00043        INSV    R2, #3, #1, (R0)                  ;
  52     61     01              1B  EF 00048        EXTZV   #27, #1, (R1), R2                 ; 0761
  60     01     04              52  F0 0004D        INSV    R2, #4, #1, (R0)                  ;
  52     61     01              1C  EF 00052        EXTZV   #28, #1, (R1), R2                 ; 0762
  60     01     05              52  F0 00057        INSV    R2, #5, #1, (R0)                  ;
  52     61     01              1D  EF 0005C        EXTZV   #29, #1, (R1), R2                 ; 0763
  60     01     06              52  F0 00061        INSV    R2, #6, #1, (R0)                  ;
                      50        01  D0 00066        MOVL    #1, R0                            ; 0766
```

```
                          04 00069 2$:     RET                          ; 0767

; Routine Size:  106 bytes,    Routine Base:  _LIB$CODE + 0069

;  515          0768  1 !<BLF/PAGE>
```

```
;   517     0769  1 %SBTTL 'SCR$SET_OUTPUT - Establish Terminal for Output'
;   518     0770  1 GLOBAL ROUTINE SCR$SET_OUTPUT (
;   519     0771  1                                 STREAM       :     VECTOR [,WORD],
;   520     0772  1                                 FILE_SPEC    : REF BLOCK [,BYTE],
;   521     0773  1                                 USER_ROUTINE : REF VECTOR [,LONG],
;   522     0774  1                                 USER_ARG     : REF VECTOR [,LONG],
;   523     0775  1                                 OLD_STREAM   : REF VECTOR [,WORD]
;   524     0776  1                               ) =
;   525     0777  1 !++
;   526     0778  1 ! FUNCTIONAL DESCRIPTION:
;   527     0779  1 !
;   528     0780  1 !       This routine sets up a device to receive output.  If this is
;   529     0781  1 !       the first call, obtain device characteristics.
;   530     0782  1 !
;   531     0783  1 !       If this is the first call for the screen than a screen
;   532     0784  1 !       control block is allocated, a channel is assigned, device
;   533     0785  1 !       characteristics are obtained.  If it is an unknown device then
;   534     0786  1 !       the channel is deassigned (no QIO output).  If a file
;   535     0787  1 !       specification is present and no user routine is declared then
;   536     0788  1 !       the file is opened via RMS.
;   537     0789  1 !
;   538     0790  1 !       At output time the user-supplied routine, if present, will be
;   539     0791  1 !       called with the following parameters:
;   540     0792  1 !
;   541     0793  1 !               AP -->  4
;   542     0794  1 !                       Optional user supplied argument.
;   543     0795  1 !                       Address of channel number (0 if none)
;   544     0796  1 !                       Address of string dsc to output
;   545     0797  1 !                       Address of stream number
;   546     0798  1 !
;   547     0799  1 ! CALLING SEQUENCE:
;   548     0800  1 !
;   549     0801  1 !       ret_status.wlc.v = SCR$SET_OUTPUT (STREAM.rw.v
;   550     0802  1 !                                          [,FILE_SPEC.rt.dx
;   551     0803  1 !                                          [,USER_ROUTINE.zem.rp
;   552     0804  1 !                                          [,USER_ARG.rl.r
;   553     0805  1 !                                          [,OLD_STREAM.wl.r]]]] )
;   554     0806  1 !
;   555     0807  1 ! FORMAL PARAMETERS:
;   556     0808  1 !
;   557     0809  1 !       STREAM.rw.v             Stream number.
;   558     0810  1 !
;   559     0811  1 !       FILE_SPEC.rt.dx         Optional.  Address of descriptor of
;   560     0812  1 !                               file specification.
;   561     0813  1 !
;   562     0814  1 !       USER_ROUTINE.zem.rp     Optional.  Address of output routine to
;   563     0815  1 !                               call for output.
;   564     0816  1 !
;   565     0817  1 !       USER_ARG.rl.r           Optional.  Address of argument to be
;   566     0818  1 !                               passed to user-specified output routine.
;   567     0819  1 !
;   568     0820  1 !       OLD_STREAM.wt.dx        Optional.  Address of longword to
;   569     0821  1 !                               receive previous stream.
;   570     0822  1 !
;   571     0823  1 ! IMPLICIT INPUTS:
;   572     0824  1 !
;   573     0825  1 !       NONE
```

```
;  574          0826  1 !
;  575          0827  1 ! IMPLICIT OUTPUTS:
;  576          0828  1 !
;  577          0829  1 !       NONE
;  578          0830  1 !
;  579          0831  1 ! COMPLETION STATUS:
;  580          0832  1 !
;  581          0833  1 !       SS$_NORMAL      Normal successful completion
;  582          0834  1 !
;  583          0835  1 ! SIDE EFFECTS:
;  584          0836  1 !
;  585          0837  1 !       NONE
;  586          0838  1 !--
;  587          0839  1
;  588          0840  2     BEGIN
;  589          0841  2
;  590          0842  2     BUILTIN
;  591          0843  2         NULLPARAMETER;
;  592          0844  2
;  593          0845  2 ! $FIND_TCB
;  594          0846  2 ! This macro searches down the threaded list of TCB's trying to find
;  595          0847  2 ! the one whose SCR$L_STREAM field matches STREAM.  If found, FOUND
;  596          0848  2 ! is set to the matching TCB address.  If we run off end of threaded
;  597          0849  2 ! list before finding match, FOUND is set to 0.
;  598          0850  2 !-
;  599        M 0851  2 MACRO    $FIND_TCB (FOUND) =
;  600        M 0852  2     BEGIN
;  601        M 0853  2     NEXT = .SCR$L_FLINKHEAD ;           ! Initialize to 1st
;  602        M 0854  2
;  603        M 0855  2 !+
;  604        M 0856  2 ! Search down chain of TCB's until we reach the one whose
;  605        M 0857  2 ! SCR$L_STREAM field matches STREAM or reach end of chain
;  606        M 0858  2 ! (indicated by a zero forward link).
;  607        M 0859  2 !-
;  608        M 0860  2     FOUND = 0 ;           ! Init to not-found
;  609        M 0861  2     WHILE .NEXT NEQ 0
;  610        M 0862  2     DO
;  611        M 0863  2         BEGIN             ! Search for desired TCB or end of list
;  612        M 0864  2         IF .NEXT [SCR$L_STREAM] EQL .STREAM [0]
;  613        M 0865  2         THEN
;  614        M 0866  2             BEGIN
;  615        M 0867  2             FOUND = .NEXT ;
;  616        M 0868  2             EXITLOOP ;  ! Break out of search loop
;  617        M 0869  2             END;
;  618        M 0870  2
;  619        M 0871  2         NEXT = .NEXT [SCR$L_FLINK] ; ! Advance pointer down chain
;  620        M 0872  2         END ;             ! Search for desired TCB or end of list
;  621        M 0873  2
;  622        M 0874  2     END;
;  623          0875  2             % ; ! End of macro $FIND_TCB
;  624          0876  2
;  625          0877  2     LOCAL
;  626          0878  2         FOUND : REF BLOCK [, BYTE],     ! Pointer to Terminal Control
;  627          0879  2                                         ! block used by $FIND_TCB
;  628          0880  2         NEXT : REF BLOCK [,BYTE],       ! Pointer to Terminal Control
;  629          0881  2                                         ! block used by $FIND_TCB
;  630          0882  2         TCB : REF BLOCK [, BYTE] ;      ! Pointer to a Terminal Control
```

```
;  631     0883  2                                          ! block
;  632     0884  2
;  633     0885  2      IF .SCR$L_CUROUTPUT EQL 0   ! If no current TCB
;  634     0886  2      THEN
;  635     0887  3          BEGIN   ! No current TCB
;  636     0888  3          $FIND_TCB (FOUND) ;
;  637     0889  3          END     ! No current TCB
;  638     0890  2      ELSE
;  639     0891  3          BEGIN   ! Current TCB exists
;  640     0892  3          TCB = .SCR$L_CUROUTPUT ;
;  641     0893  3          FOUND = .TCB ;
;  642     0894  3          IF .TCB [SCR$L_STREAM] NEQ .STREAM [0]
;  643     0895  3          THEN
;  644     0896  4              BEGIN       ! Not the one we want
;  645     0897  4              $FIND_TCB (FOUND) ;
;  646     0898  3              END;        ! Not the one we want
;  647     0899  2          END;    ! Current TCB exists
;  648     0900  2
;  649     0901  2  !+
;  650     0902  2  ! Reach here when we have found desired TCB or have exhausted chain.
;  651     0903  2  ! Decide which, and treat appropriately.
;  652     0904  2  !-
;  653     0905  2      IF .FOUND EQL 0
;  654     0906  2      THEN
;  655     0907  3          BEGIN   ! Ran off end of list, must build new TCB
;  656     0908  3          LOCAL
;  657     0909  3              TYPE,                   ! Device type returned by GET_CHAR
;  658     0910  3              AREA,                   ! pagesize * page width
;  659     0911  3              TOT_SPACE,              ! Total buffer space to get from GET_VM
;  660     0912  3              STATUS,                 ! Status for subroutine calls --
;  661     0913  3                                      ! returned to user if not success
;  662     0914  3
;  663     0915  3              LOC_DESC : BLOCK [8, BYTE] ; ! Local fixed-length string
;  664     0916  3                                      ! descriptor.
;  665     0917  3          !+
;  666     0918  3          ! If exit handler hasn't been established, do it now.
;  667     0919  3          !-
;  668     0920  3          IF .SCR$EXITBLOCK[1] EQL 0
;  669     0921  3          THEN
;  670     0922  4              BEGIN       ! Set up exit handler
;  671     0923  4              LOCAL
;  672     0924  4                  STATUS ; ! Locally used status
;  673     0925  4              SCR$EXITBLOCK [1] = EXIT_HANDLER ;
;  674     0926  4              SCR$EXITBLOCK [3] = SCR$C_EXITSTS ;
;  675     0927  5              IF NOT (STATUS = $DCLEXH ( DESBLK = SCR$EXITBLOCK[0]))
;  676     0928  4              THEN
;  677     0929  4                  RETURN (.STATUS );
;  678     0930  3              END;        ! Set up exit handler
;  679     0931  3
;  680     0932  3          !+
;  681     0933  3          ! Allocate space for a new TCB
;  682     0934  3          !-
;  683     0935  4          IF NOT (STATUS = LIB$GET_VM ( %REF (SCR$C_SIZE), TCB ))
;  684     0936  3          THEN
;  685     0937  3              RETURN (.STATUS) ;
;  686     0938  3
;  687     0939  3          !+
```

```
;  688     0940  3          ! Clear new TCB to zero and link it into our chain of TCB's
;  689     0941  3          !-
;  690     0942  3          CH$FILL ( 0, SCR$C_SIZE, TCB [SCR$L_FLINK]) ;
;  691     0943  3          TCB [SCR$L_FLINK] = .SCR$L_FLINKHEAD ;   ! This TCB's forward
;  692     0944  3                                                   ! pointer = current
;  693     0945  3                                                   ! list header contents
;  694     0946  3          SCR$L_FLINKHEAD = TCB [SCR$L_FLINK] ;    ! List head points to
;  695     0947  3                                                   ! new TCB
;  696     0948  3
;  697     0949  3          TCB [SCR$L_STREAM] = .STREAM [0] ;       ! Plug in stream id
;  698     0950  3
;  699     0951  3          TCB [SCR$L_BUFSIZ] = BUFSIZE ;  ! Buffer size we'll use
;  700     0952  3
;  701     0953  3          !+
;  702     0954  3          ! Set up optional arguments
;  703     0955  3          !-
;  704     0956  3          IF NOT NULLPARAMETER (5)
;  705     0957  3          THEN
;  706     0958  4              OLD_STREAM [0] = (IF .SCR$L_CUROUTPUT EQL 0
;  707     0959  4                                THEN 0
;  708     0960  3                                ELSE .SCR$L_CUROUTPUT [SCR$L_STREAM]) ;
;  709     0961  3
;  710     0962  3          SCR$L_CUROUTPUT = .TCB ; ! Establish this TCB as current
;  711     0963  3
;  712     0964  4          TCB [SCR$L_RTNADDR] = ( IF NULLPARAMETER (3)
;  713     0965  4                                  THEN 0
;  714     0966  3                                  ELSE USER_ROUTINE [0] ) ;
;  715     0967  3
;  716     0968  4          TCB [SCR$L_RTNARG]  = ( IF NULLPARAMETER (4)
;  717     0969  4                                  THEN 0
;  718     0970  3                                  ELSE USER_ARG [0] ) ;
;  719     0971  3
;  720     0972  3          !+
;  721     0973  3          ! If user supplied a file spec use it, else use default filespec
;  722     0974  3          ! of SYS$OUTPUT.
;  723     0975  3          !-
;  724     0976  3          LOC_DESC [DSC$B_CLASS] = DSC$K_CLASS_S ;
;  725     0977  3          LOC_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T ;
;  726     0978  3          IF NOT NULLPARAMETER (2)
;  727     0979  3          THEN
;  728     0980  4              BEGIN       ! User-supplied file spec
;  729     0981  5              IF NOT (STATUS =
;  730     0982  5                  LIB$ANALYZE_SDESC_R2 ( .FILE_SPEC ;
;  731     0983  5                                         LOC_DESC [DSC$W_LENGTH],
;  732     0984  5                                         LOC_DESC [DSC$A_POINTER] ))
;  733     0985  4              THEN
;  734     0986  4                  RETURN (.STATUS ) ;
;  735     0987  4              END         ! User-supplied file spec
;  736     0988  3          ELSE
;  737     0989  4              BEGIN       ! Use default file spec
;  738     0990  4              LOC_DESC [DSC$W_LENGTH]  = %CHARCOUNT   ('SYS$OUTPUT') ;
;  739     0991  4              LOC_DESC [DSC$A_POINTER] = UPLIT ( BYTE ('SYS$OUTPUT')) ;
;  740     0992  3              END;        ! Use default file spec
;  741     0993  3
;  742     0994  3          !+
;  743     0995  3          ! Assign a channel
;  744     0996  3          !-
```

```
;  745     0997  4          IF NOT (STATUS = LIB$ASSIGN ( LOC_DESC, TCB [SCR$W_CHAN] ))
;  746     0998  4          THEN
;  747     0999  4              RETURN (.STATUS) ;
;  748     1000  4
;  749     1001  4          !+
;  750     1002  4          ! Determine type of terminal and if known type get a local
;  751     1003  4          ! event flag number to use while doing QIO's to it.  If we
;  752     1004  4          ! can't get an event flag number, quit.
;  753     1005  4          !-
;  754     1006  4          GET_CHAR ( .TCB, TYPE ) ;
;  755     1007  4          IF .TYPE NEQ 0
;  756     1008  4          THEN
;  757     1009  4              BEGIN ! Known type
;  758     1010  4              LOCAL
;  759     1011  4                  STATUS ;        ! Local status
;  760     1012  4
;  761     1013  5              IF NOT (STATUS = LIB$GET_EF ( TCB [SCR$L_EFN]))
;  762     1014  4              THEN
;  763     1015  4                  RETURN (.STATUS ) ;
;  764     1016  4
;  765     1017  4              RETURN ( SS$_NORMAL) ;
;  766     1018  4              END;  ! Known type
;  767     1019  4
;  768     1020  4          !+
;  769     1021  4          ! Reach here if unknown terminal or not a terminal -- deassign
;  770     1022  4          ! QIO channel.
;  771     1023  4          !-
;  772     1024  4          IF NOT (STATUS = $DASSGN ( CHAN = .TCB [SCR$W_CHAN]))
;  773     1025  4          THEN
;  774     1026  4              RETURN (.STATUS ) ;
;  775     1027  4
;  776     1028  4          TCB [SCR$W_CHAN] = 0 ;  ! Clear channel number
;  777     1029  4
;  778     1030  3          IF .TCB [SCR$L_RTNADDR] EQL 0
;  779     1031  3          THEN
;  780     1032  4              BEGIN       ! No user-specified call back
;  781     1033  4              IF NOT NULLPARAMETER (2)
;  782     1034  4              THEN
;  783     1035  5                  BEGIN   ! Filespec supplied
;  784     1036  5                  LOCAL
;  785     1037  5                      LENGTH,     ! returned by LIB$ANALYZE_SDESC_R2
;  786     1038  5                      ADDR,       ! addr. returned by LIB$ANALYZE_SDESC_R2
;  787     1039  5                      STATUS ;    ! Local status
;  788     1040  6                  IF NOT (STATUS = LIB$ANALYZE_SDESC_R2 ( .FILE_SPEC ;
;  789     1041  6                                                          LENGTH,
;  790     1042  6                                                          ADDR ))
;  791     1043  5                  THEN
;  792     1044  5                      RETURN ( .STATUS ) ;
;  793     1045  5
;  794     1046  6                  IF NOT (STATUS = CREATE ( TCB [SCR$L_FLINK], LENGTH,
;  795     1047  6                                            ADDR ))
;  796     1048  5                  THEN
;  797     1049  5                      RETURN (.STATUS ) ;
;  798     1050  4                  END ;   ! Filespec supplied
;  799     1051  3              END ;       ! No user-specified call back
;  800     1052  3
;  801     1053  3
```

```
                !+
                ! Allocate and initialize map buffers necessary to emulate
                ! advanced VDT features when outputing to hardcopy, disk, or
                ! limited VDT's.
                !
                ! A contiguous space composed of the following parts is allocated:
                !       Name                    Size
                !   character map          area = device length * width
                !   attribute map          area
                !   modified map           area/8 + 1 (bit map size in bytes)
                !-
                AREA = .TCB [SCR$W_DEVPAGSIZ] * .TCB [SCR$W_DEVWIDTH] ;
                TOT_SPACE = (.AREA/%BPUNIT) + 1 + (2 * .AREA) ;
                IF NOT (STATUS = LIB$GET_VM ( TOT_SPACE, TCB [SCR$L_CHARMAP] ))
                THEN
                    RETURN ( .STATUS ) ;

                TCB [SCR$L_AREA] = .AREA ;
                CH$FILL (%C' ', .AREA, .TCB [SCR$L_CHARMAP]) ;
                                        ! space fill character map
                TCB [SCR$L_ATTRMAP] = .TCB [SCR$L_CHARMAP] + .AREA ;
                TCB [SCR$L_MODFMAP] = .TCB [SCR$L_CHARMAP] + (2 * .AREA) ;
                CH$FILL (0, .TOT_SPACE - .AREA, .TCB [SCR$L_CHARMAP] + .AREA) ;
                                        ! zero fill attrmap and modfmap
                TCB [SCR$L_LINE] = 1 ;
                TCB [SCR$L_COLUMN] = 1 ;
                END     ! Ran off end of list, must build new TCB

        ELSE

            BEGIN   ! Found desired TCB
            !+
            ! Reach here when we have found the TCB we want.
            !-
            IF NOT NULLPARAMETER (5)
            THEN
                OLD_STREAM [0] = .SCR$L_CUROUTPUT [SCR$L_STREAM] ;

            TCB = .FOUND ;   ! Record which one found
            SCR$L_CUROUTPUT = .TCB ;

            TCB [SCR$L_RTNADDR] = ( IF NULLPARAMETER (3)
                                    THEN 0
                                    ELSE USER_ROUTINE [0] ) ;

            TCB [SCR$L_RTNARG]  = ( IF NULLPARAMETER (4)
                                    THEN 0
                                    ELSE USER_ARG [0] ) ;
            END;    ! Found desired TCB

        RETURN (SS$_NORMAL);
        END;                                    ! End of routine SCR$SET_OUTPUT


                                         000D3            .BLKB   1
   54 55 50 54 55 4F 24 53 59 53         000D4 P.AAA:   .ASCII  \SYS$OUTPUT\                        ;
```

```
                                                                .EXTRN  SYS$DCLEXH, SYS$DASSGN

                                                        0FFC 00000      .ENTRY  SCR$SET_OUTPUT, Save R2,R3,R4,R5,R6,R7,R8,- ; 0770
                                                                                R9,R10,R11
                                                5B 00000000G   00  9E 00002      MOVAB   SCR$L_CUROUTPUT, R11
                                                5A 00000000G   00  9E 00009      MOVAB   SCR$L_FLINKHEAD, R10
                                                59 00000000'   EF  9E 00010      MOVAB   SCR$EXITBLOCK+4, R9
                                                5E                 20  C2 00017      SUBL2   #32, SP
                                                52                 6B  D0 0001A      MOVL    SCR$L_CUROUTPUT, R2                     ; 0885
                                                                   17  12 0001D      BNEQ    2$
                                                50                 6A  D0 0001F      MOVL    SCR$L_FLINKHEAD, NEXT                   ; 0888
                                                                   53  D4 00022      CLRL    FOUND
                                                                   50  D5 00024 1$:  TSTL    NEXT
                                                                   3E  13 00026      BEQL    6$
    30   A0        04   AC                      10                 00  ED 00028      CMPZV   #0, #16, STREAM, 48(NEXT)
                                                                   2B  13 0002F      BEQL    4$
                                                50                 60  D0 00031      MOVL    (NEXT), NEXT
                                                                   EE  11 00034      BRB     1$
                                          04    AE                 52  D0 00036 2$:  MOVL    R2, TCB                                 ; 0892
                                                51        04       AE  D0 0003A      MOVL    TCB, R1                                 ; 0893
                                                53                 51  D0 0003E      MOVL    R1, FOUND
    30   A1        04   AC                      10                 00  ED 00041      CMPZV   #0, #16, STREAM, 48(R1)                 ; 0894
                                                                   1C  13 00048      BEQL    6$
                                                50                 6A  D0 0004A      MOVL    SCR$L_FLINKHEAD, NEXT                   ; 0897
                                                                   53  D4 0004D      CLRL    FOUND
                                                                   50  D5 0004F 3$:  TSTL    NEXT
                                                                   13  13 00051      BEQL    6$
    30   A0        04   AC                      10                 00  ED 00053      CMPZV   #0, #16, STREAM, 48(NEXT)
                                                                   05  12 0005A      BNEQ    5$
                                                53                 50  D0 0005C 4$:  MOVL    NEXT, FOUND
                                                                   05  11 0005F      BRB     6$
                                                50                 60  D0 00061 5$:  MOVL    (NEXT), NEXT
                                                                   E9  11 00064      BRB     3$
                                                                   53  D5 00066 6$:  TSTL    FOUND                                   ; 0905
                                                                   03  13 00068      BEQL    7$
                                                                 01A9  31 0006A      BRW     28$
                                                                   69  D5 0006D 7$:  TSTL    SCR$EXITBLOCK+4                         ; 0920
                                                                   18  12 0006F      BNEQ    8$
                                                69       0000V     CF  9E 00071      MOVAB   EXIT_HANDLER, SCR$EXITBLOCK+4           ; 0925
                                          08    A9         0C     A9  9E 00076      MOVAB   SCR$L_EXITSTS, SCR$EXITBLOCK+12         ; 0926
                                                             FC     A9  9F 0007B      PUSHAB  SCR$EXITBLOCK                           ; 0927
                                     00000000G  00                 01  FB 0007E      CALLS   #1, SYS$DCLEXH
                                                01                 50  E8 00085      BLBS    STATUS, 8$
                                                                       04 00088      RET
                                                             04    AE  9F 00089 8$:  PUSHAB  TCB                                     ; 0935
                                          04    AE                 78  8F 9A 0008C      MOVZBL  #120, 4(SP)
                                                             04    AE  9F 00091      PUSHAB  4(SP)
                                     00000000G  00                 02  FB 00094      CALLS   #2, LIB$GET_VM
                                                58                 50  D0 0009B      MOVL    R0, STATUS
                                                03                 58  E8 0009E      BLBS    STATUS, 9$
                                                                 0140  31 000A1      BRW     26$
                                                57        04       AE  D0 000A4 9$:  MOVL    TCB, R7                                 ; 0942
  0078   8F              00                     6E                 00  2C 000A8      MOVC5   #0, (SP), #0, #120, (R7)
                                                                   67      000AF
                                                67                 6A  D0 000B0      MOVL    SCR$L_FLINKHEAD, (R7)                   ; 0943
                                                6A                 57  D0 000B3      MOVL    R7, SCR$L_FLINKHEAD                     ; 0946
```

```
          30   A7        04   AC  3C 000B6         MOVZWL   STREAM, 48(R7)                  ; 0949
          4C   A7      0200   8F  3C 000BB         MOVZWL   #512, 76(R7)                    ; 0951
               05             6C  91 000C1         CMPB     (AP), #5                        ; 0956
                              16  1F 000C4         BLSSU    12$                             ;
                         14   AC  D5 000C6         TSTL     20(AP)                          ;
                              11  13 000C9         BEQL     12$                             ;
               50             6B  D0 000CB         MOVL     SCR$L_CUROUTPUT, R0             ; 0958
                              04  12 000CE         BNEQ     10$                             ;
                              50  D4 000D0         CLRL     R0                              ;
                              04  11 000D2         BRB      11$                             ;
               50        30   A0  D0 000D4  10$:   MOVL     48(R0), R0                      ; 0960
          14   BC             50  B0 000D8  11$:   MOVW     R0, @OLD_STREAM                 ; 0958
               6B             57  D0 000DC  12$:   MOVL     R7, SCR$L_CUROUTPUT             ; 0962
               03             6C  91 000DF         CMPB     (AP), #3                        ; 0964
                              05  1F 000E2         BLSSU    13$                             ;
                         0C   AC  D5 000E4         TSTL     12(AP)                          ;
                              04  12 000E7         BNEQ     14$                             ;
                              50  D4 000E9  13$:   CLRL     R0                              ;
                              04  11 000EB         BRB      15$                             ;
               50        0C   AC  D0 000ED  14$:   MOVL     USER_ROUTINE, R0                ; 0966
          38   A7             50  D0 000F1  15$:   MOVL     R0, 56(R7)                      ; 0964
               04             6C  91 000F5         CMPB     (AP), #4                        ; 0968
                              05  1F 000F8         BLSSU    16$                             ;
                         10   AC  D5 000FA         TSTL     16(AP)                          ;
                              04  12 000FD         BNEQ     17$                             ;
                              50  D4 000FF  16$:   CLRL     R0                              ;
                              04  11 00101         BRB      18$                             ;
               50        10   AC  D0 00103  17$:   MOVL     USER_ARG, R0                    ; 0970
          3C   A7             50  D0 00107  18$:   MOVL     R0, 60(R7)                      ; 0968
          1A   AE      010E   8F  B0 0010B         MOVW     #270, LOC_DESC+2                ; 0977
               02             6C  91 00111         CMPB     (AP), #2                        ; 0978
                              20  1F 00114         BLSSU    20$                             ;
                         08   AC  D5 00116         TSTL     8(AP)                           ;
                              1B  13 00119         BEQL     20$                             ;
               50        08   AC  D0 0011B         MOVL     FILE_SPEC, R0                   ; 0983
                  00000000G   00  16 0011F         JSB      LIB$ANALYZE_SDESC_R2            ;
               58             50  D0 00125         MOVL     R0, STATUS                      ;
          18   AE             51  B0 00128         MOVW     R1, LOC_DESC                    ;
          1C   AE             52  D0 0012C         MOVL     R2, LOC_DESC+4                  ; 0984
               0D             58  E8 00130         BLBS     STATUS, 21$                     ; 0983
                       00AE   31     00133  19$:   BRW      26$                             ; 0986
          18   AE             0A  B0 00136  20$:   MOVW     #10, LOC_DESC                   ; 0990
          1C   AE      FEB6   CF  9E 0013A         MOVAB    P.AAA, LOC_DESC+4               ; 0991
                         08   A7  9F 00140  21$:   PUSHAB   8(R7)                           ; 0997
                         1C   AE  9F 00143         PUSHAB   LOC_DESC                        ;
   00000000G   00             02  FB 00146         CALLS    #2, LIB$ASSIGN                  ;
               58             50  D0 0014D         MOVL     R0, STATUS                      ;
               E0             58  E9 00150         BLBC     STATUS, 19$                     ;
                         08   AE  9F 00153         PUSHAB   TYPE                            ; 1006
                              57  DD 00156         PUSHL    R7                              ;
       0000V   CF             02  FB 00158         CALLS    #2, GET_CHAR                    ;
                         08   AE  D5 0015D         TSTL     TYPE                            ; 1007
                              11  13 00160         BEQL     23$                             ;
                         48   A7  9F 00162         PUSHAB   72(R7)                          ; 1013
   00000000G   00             01  FB 00165         CALLS    #1, LIB$GET_EF                  ;
               03             50  E9 0016C         BLBC     STATUS, 22$                     ;
                       00EA   31     0016F         BRW      36$                             ;
```

```
                                        04 00172 22$:   RET                                             ; 1017
                         7E       08    A7 3C 00173 23$: MOVZWL  8(R7), -(SP)                           ; 1024
             00000000G   00             01 FB 00177      CALLS   #1, SYS$DASSGN
                         58             50 D0 0017E      MOVL    R0, STATUS
                         60             58 E9 00181      BLBC    STATUS, 26$
                                  08    A7 B4 00184      CLRW    8(R7)                                  ; 1028
                                  38    A7 D5 00187      TSTL    56(R7)                                 ; 1030
                                        30 12 0018A      BNEQ    25$
                         02             6C 91 0018C      CMPB    (AP), #2                               ; 1033
                                        2B 1F 0018F      BLSSU   25$
                                  08    AC D5 00191      TSTL    8(AP)
                                        26 13 00194      BEQL    25$
                         50       08    AC D0 00196      MOVL    FILE_SPEC, R0                          ; 1040
                     00000000G    00    16 0019A         JSB     LIB$ANALYZE_SDESC_R2
                   10    AE             51 D0 001A0      MOVL    R1, LENGTH
                   0C    AE             52 D0 001A4      MOVL    R2, ADDR
                         0D             50 E9 001A8      BLBC    STATUS, 24$
                                  0C    AE 9F 001AB      PUSHAB  ADDR                                   ; 1046
                                  14    AE 9F 001AE      PUSHAB  LENGTH
                                        57 DD 001B1      PUSHL   R7
                 0000V   CF             03 FB 001B3      CALLS   #3, CREATE
                         01             50 E8 001B8 24$: BLBS    STATUS, 25$
                                        04 001BB         RET
                         56       0E    A7 3C 001BC 25$: MOVZWL  14(R7), AREA                           ; 1065
                         50       0C    A7 3C 001C0      MOVZWL  12(R7), R0
                         56             50 C4 001C4      MULL2   R0, AREA
                   50    56             08 C7 001C7      DIVL3   #8, AREA, R0                           ; 1066
                   14    AE       01 A046 3E 001CB       MOVAW   1(R0)[AREA], TOT_SPACE
                                  18    A7 9F 001D1      PUSHAB  24(R7)                                 ; 1067
                                  18    AE 9F 001D4      PUSHAB  TOT_SPACE
             00000000G   00             02 FB 001D7      CALLS   #2, LIB$GET_VM
                         58             50 D0 001DE      MOVL    R0, STATUS
                         04             58 E8 001E1      BLBS    STATUS, 27$
                         50             58 D0 001E4 26$: MOVL    STATUS, R0                             ; 1069
                                        04 001E7         RET
                   14    A7             56 D0 001E8 27$: MOVL    AREA, 20(R7)                           ; 1071
       56          20    6E             00 2C 001EC      MOVC5   #0, (SP), #32, AREA, @24(R7)           ; 1072
                                  18    B7    001F1
                   1C    A7       18 B746 9E 001F3       MOVAB   @24(R7)[AREA], 28(R7)                  ; 1074
                   20    A7       18 B746 3E 001F9       MOVAW   @24(R7)[AREA], 32(R7)                  ; 1075
                   50    14    AE       56 C3 001FF      SUBL3   AREA, TOT_SPACE, R0                    ; 1076
       50          00    6E             00 2C 00204      MOVC5   #0, (SP), #0, R0, @24(R7)[AREA]
                                  18 B746     00209
                   24    A7             01 D0 0020C      MOVL    #1, 36(R7)                             ; 1078
                   28    A7             01 D0 00210      MOVL    #1, 40(R7)                             ; 1079
                                        46 11 00214      BRB     36$                                    ; 0905
                         05             6C 91 00216 28$: CMPB    (AP), #5                               ; 1088
                                        0A 1F 00219      BLSSU   29$
                                  14    AC D5 0021B      TSTL    20(AP)
                                        05 13 0021E      BEQL    29$
                   14    BC       30    A2 B0 00220      MOVW    48(R2), @OLD_STREAM                    ; 1090
                   04    AE             53 D0 00225 29$: MOVL    FOUND, TCB                             ; 1092
                         50       04    AE D0 00229      MOVL    TCB, R0                                ; 1093
                         6B             50 D0 0022D      MOVL    R0, SCR$L_CUROUTPUT
                         03             6C 91 00230      CMPB    (AP), #3                               ; 1095
                                        05 1F 00233      BLSSU   30$
                                  0C    AC D5 00235      TSTL    12(AP)
```

```
                        04  12 00238          BNEQ    31$                         ;
                        51  D4 0023A 30$:     CLRL    R1                          ;
                        04  11 0023C          BRB     32$                         ;
          51      0C    AC  D0 0023E 31$:     MOVL    USER_ROUTINE, R1            ; 1097
     38   A0            51  D0 00242 32$:     MOVL    R1, 56(R0)                  ; 1095
          04            6C  91 00246          CMPB    (AP), #4                    ; 1099
                        05  1F 00249          BLSSU   33$                         ;
                  10    AC  D5 0024B          TSTL    16(AP)                      ;
                        04  12 0024E          BNEQ    34$                         ;
                        51  D4 00250 33$:     CLRL    R1                          ;
                        04  11 00252          BRB     35$                         ;
          51      10    AC  D0 00254 34$:     MOVL    USER_ARG, R1                ; 1101
     3C   A0            51  D0 00258 35$:     MOVL    R1, 60(R0)                  ; 1099
          50            01  D0 0025C 36$:     MOVL    #1, R0                      ; 1104
                            04 0025F          RET                                 ; 1105

; Routine Size:  608 bytes,    Routine Base:  _LIB$CODE + 00DE

;  854          1106  1 !<BLF/PAGE>
```

```
;  856     1107  1 %SBTTL 'SCR$STOP_OUTPUT - Stop Output to Terminal or Screen Buffer'
;  857     1108  1 GLOBAL ROUTINE SCR$STOP_OUTPUT =
;  858     1109  1 !++
;  859     1110  1 ! FUNCTIONAL DESCRIPTION:
;  860     1111  1 !
;  861     1112  1 !       This routine deaccesses current stream established for output.
;  862     1113  1 !
;  863     1114  1 ! CALLING SEQUENCE:
;  864     1115  1 !
;  865     1116  1 !       ret_status.wlc.v = SCR$STOP_OUTPUT ( )
;  866     1117  1 !
;  867     1118  1 ! FORMAL PARAMETERS:
;  868     1119  1 !
;  869     1120  1 !       NONE
;  870     1121  1 !
;  871     1122  1 ! IMPLICIT INPUTS:
;  872     1123  1 !
;  873     1124  1 !       NONE
;  874     1125  1 !
;  875     1126  1 ! IMPLICIT OUTPUTS:
;  876     1127  1 !
;  877     1128  1 !       NONE
;  878     1129  1 !
;  879     1130  1 ! COMPLETION STATUS:
;  880     1131  1 !
;  881     1132  1 !       SS$_NORMAL      Normal successful completion
;  882     1133  1 !
;  883     1134  1 ! SIDE EFFECTS:
;  884     1135  1 !
;  885     1136  1 !       The channel or ISI is deassigned.
;  886     1137  1 !--
;  887     1138  1
;  888     1139  2     BEGIN
;  889     1140  2     LOCAL
;  890     1141  2         STATUS,                             ! Status to return to caller
;  891     1142  2         STATUS2,                            ! Temporary internal status
;  892     1143  2         LAST : REF BLOCK [, BYTE],          ! Pointer to a Terminal Control
;  893     1144  2                                             ! block
;  894     1145  2         NEXT : REF BLOCK [, BYTE] ;         ! Pointer to a Terminal Control
;  895     1146  2                                             ! block
;  896     1147  2
;  897     1148  2     LAST = SCR$L_FLINKHEAD ;                ! Initialize to head of chain
;  898     1149  2     NEXT = .SCR$L_FLINKHEAD ;               ! Initialize to 1st
;  899     1150  2
;  900     1151  2 !+
;  901     1152  2 ! Search down chain of TCB's until we reach the one that matches
;  902     1153  2 ! SCR$L_CUROUTPUT or reach end of chain (indicated by a zero forward
;  903     1154  2 ! link).
;  904     1155  2 !-
;  905     1156  2     WHILE .NEXT NEQ 0
;  906     1157  2     DO
;  907     1158  3         BEGIN           ! Search for desired TCB or end of list
;  908     1159  3         IF .NEXT EQL .SCR$L_CUROUTPUT
;  909     1160  3         THEN
;  910     1161  3             EXITLOOP ;  ! Break out of search loop
;  911     1162  3
;  912     1163  3         LAST = .NEXT ;  ! Remember last entry address
```

```
;  913          1164  3          NEXT = .NEXT [SCR$L_FLINK] ; ! Advance pointer down chain
;  914          1165  3          END ;              ! Search for desired TCB or end of list
;  915          1166  2
;  916          1167  2      !+
;  917          1168  2      ! Reach here when we have found desired TCB or have exhausted chain.
;  918          1169  2      ! Decide which, and treat appropriately.
;  919          1170  2      !-
;  920          1171  2          IF .NEXT EQL 0
;  921          1172  2          THEN
;  922          1173  3              BEGIN   ! Ran off end of list
;  923          1174  3              STATUS = SS$_NORMAL ;
;  924          1175  3              END     ! Ran off end of list
;  925          1176  2
;  926          1177  2          ELSE
;  927          1178  2
;  928          1179  3              BEGIN   ! Located TCB we want
;  929          1180  3              !+
;  930          1181  3              ! First order of business is to remove this TCB from chain.
;  931          1182  3              ! Set previous entry's forward pointer to the contents of this
;  932          1183  3              ! entry's forward pointer.
;  933          1184  3              !-
;  934          1185  3              LAST [SCR$L_FLINK] = .NEXT [SCR$L_FLINK] ;
;  935          1186  3
;  936          1187  3              !+
;  937          1188  3              ! If a there is a channel involved ?
;  938          1189  3              !-
;  939          1190  3              IF .NEXT [SCR$W_CHAN] NEQ 0
;  940          1191  3              THEN
;  941          1192  4                  BEGIN       ! Channel involved
;  942          1193  4                  NEXT [SCR$L_BUFFER] = 0 ;   ! Turn off buffering
;  943          1194  4
;  944          1195  4                  !+
;  945          1196  4                  ! If scrolling active, turn it off.
;  946          1197  4                  !-
;  947          1198  4                  IF .NEXT [SCR$V_SCROLL] EQL 1
;  948          1199  4                  THEN
;  949          1200  5                      BEGIN ! Scrolling was on
;  950          1201  5                      NEXT [SCR$V_SCROLL] = 0 ; ! Turn off indicator
;  951          1202  5                      SCR$SET_SCROLL () ;       ! Turn off scrolling in
;  952          1203  5                                                ! terminal
;  953          1204  4                      END ; ! Scrolling was on
;  954          1205  4
;  955          1206  4                  !+
;  956          1207  4                  ! Now to deassign the channel
;  957          1208  4                  !-
;  958          1209  4                  STATUS2 = $DASSGN ( CHAN = .NEXT [SCR$W_CHAN]) ;
;  959          1210  4
;  960          1211  4                  !+
;  961          1212  4                  ! Deallocate the local Event Flag
;  962          1213  4                  !-
;  963          1214  4                  LIB$FREE_EF ( NEXT [SCR$L_EFN]) ;
;  964          1215  4
;  965          1216  3                  END ;       ! Channel involved
;  966          1217  3
;  967          1218  3          !+
;  968          1219  3          ! Check to see if a file is open
;  969          1220  3          !-
```

```
;  970          1221  3          IF .NEXT [SCR$W_IFI] NEQ 0
;  971          1222  3          THEN
;  972          1223  4              BEGIN       ! File open
;  973          1224  4              LOCAL
;  974          1225  4                  FAB : REF $FAB_DECL;    ! ptr to FAB
;  975          1226  4
;  976          1227  4              FAB = .NEXT [SCR$L_FAB];
;  977          1228  4
;  978          1229  4              CH$FILL (0, FAB$C_BLN, .FAB ) ; ! Clear FAB to zero
;  979          1230  4              FAB [FAB$W_IFI] = .NEXT [SCR$W_IFI] ;   ! File IFI
;  980          1231  4              FAB [FAB$B_BID] = FAB$C_BID ;           ! Identify block as FAB
;  981          1232  4              FAB [FAB$B_BLN] = FAB$C_BLN ;           ! Length of FAB
;  982          1233  4              STATUS2 = $CLOSE ( FAB = .FAB ) ; ! Close file
;  983          1234  4              IF .STATUS2
;  984          1235  4              THEN
;  985          1236  5                  BEGIN           ! free FAB and RAB space
;  986          1237  5                  LIB$FREE_VM (%REF(RAB$C_BLN), NEXT [SCR$L_RAB]);
;  987          1238  5                  LIB$FREE_VM (%REF(FAB$C_BLN), NEXT [SCR$L_FAB]);
;  988          1239  4                  END;
;  989          1240  3              END ;       ! File open
;  990          1241  3
;  991          1242  3          !+
;  992          1243  3          ! Release buffer space if we have been using a character map.
;  993          1244  3          !-
;  994          1245  3          IF .NEXT [SCR$L_CHARMAP] NEQ 0
;  995          1246  3          THEN
;  996          1247  4              BEGIN       ! Free map
;  997          1248  4              LOCAL
;  998          1249  4                  TOTAL_SPACE;
;  999          1250  4              !+
; 1000          1251  4              ! The attribute map was allocated contiguously with the
; 1001          1252  4              ! character map.  Be sure to free up both.
; 1002          1253  4              !
; 1003          1254  4              ! Space composed of the following parts was allocated:
; 1004          1255  4              !   Name                    Size
; 1005          1256  4              !  character map    area = device length * width
; 1006          1257  4              !  attribute map    area
; 1007          1258  4              !  modified map     area/8 + 1 (bit map size in bytes)
; 1008          1259  4              !-
; 1009          1260  4              TOTAL_SPACE = (.NEXT [SCR$L_AREA]/%BPUNIT) + 1 + (2 * .NEXT [SCR$L_AREA]);
; 1010          1261  4              STATUS = LIB$FREE_VM ( TOTAL_SPACE,
; 1011          1262  4                                     NEXT [SCR$L_CHARMAP] ) ;
; 1012          1263  4              IF .STATUS
; 1013          1264  4              THEN
; 1014          1265  5                  BEGIN
; 1015          1266  5                  !+
; 1016          1267  5                  ! Free the TCB area itself
; 1017          1268  5                  !-
; 1018          1269  5                  STATUS = LIB$FREE_VM ( %REF ( SCR$C_SIZE),      ! length
; 1019          1270  5                                         NEXT ) ; ! base
; 1020          1271  4                  END;
; 1021          1272  3              END ;       ! Free map
; 1022          1273  3
; 1023          1274  3          !+
; 1024          1275  3          ! If status of $CLOSE was successful, return it, else
; 1025          1276  3          ! return status of LIB$FREE_VM.
; 1026          1277  3          !-
```

```
3              IF .STATUS2
3              THEN
3                  STATUS = .STATUS2 ;
3
2              END ;     ! Located TCB we want
2
2          SCR$L_CUROUTPUT = 0;
2          RETURN (.STATUS);
1          END;                                        ! End of routine SCR$STOP_OUTPUT


                                                                  .EXTRN  SYS$CLOSE

                                          0FFC 00000            .ENTRY  SCR$STOP_OUTPUT, Save R2,R3,R4,R5,R6,R7,R8,-; 1108
                                                                        R9,R10,R11                                   :
                  5B 00000000G  00  9E 00002            MOVAB   SCR$L_CUROUTPUT, R11                         :
                  5A 00000000G  00  9E 00009            MOVAB   LIB$FREE_VM, R10                             :
                  5E            0C  C2 00010            SUBL2   #12, SP                                      :
                  51 00000000G  00  9E 00013            MOVAB   SCR$L_FLINKHEAD, LAST                        ; 1148
              08  AE 00000000G  00  D0 0001A            MOVL    SCR$L_FLINKHEAD, NEXT                        ; 1149
                  50       08   AE  D0 00022 1$:        MOVL    NEXT, R0                                     ; 1156
                                0E  13 00026            BEQL    2$                                           :
                  6B            50  D1 00028            CMPL    R0, SCR$L_CUROUTPUT                          ; 1159
                                09  13 0002B            BEQL    2$                                           :
                  51            50  D0 0002D            MOVL    R0, LAST                                     ; 1163
              08  AE            60  D0 00030            MOVL    (R0), NEXT                                   ; 1164
                                EC  11 00034            BRB     1$                                           ; 1156
                  56       08   AE  D0 00036 2$:        MOVL    NEXT, R6                                     ; 1171
                                06  12 0003A            BNEQ    3$                                           :
                  58            01  D0 0003C            MOVL    #1, STATUS                                   ; 1174
                              00B1  31 0003F            BRW     8$                                           ; 1171
                  61            66  D0 00042 3$:        MOVL    (R6), (LAST)                                 ; 1185
                           08   A6  B5 00045            TSTW    8(R6)                                        ; 1190
                                2A  13 00048            BEQL    5$                                           :
                           04   A6  D4 0004A            CLRL    4(R6)                                        ; 1193
                  0B       40   A6  E9 0004D            BLBC    64(R6), 4$                                   ; 1198
              40  A6            01  8A 00051            BICB2   #1, 64(R6)                                   ; 1201
       00000000G  00            00  FB 00055            CALLS   #0, SCR$SET_SCROLL                           ; 1202
                  7E       08   A6  3C 0005C 4$:        MOVZWL  8(R6), -(SP)                                 ; 1209
       00000000G  00            01  FB 00060            CALLS   #1, SYS$DASSGN                               :
                  59            50  D0 00067            MOVL    R0, STATUS2                                  :
                           48   A6  9F 0006A            PUSHAB  72(R6)                                       ; 1214
       00000000G  00            01  FB 0006D            CALLS   #1, LIB$FREE_EF                              :
                           34   A6  B5 00074 5$:        TSTW    52(R6)                                       ; 1221
                                41  13 00077            BEQL    6$                                           :
                  57       70   A6  D0 00079            MOVL    112(R6), FAB                                 ; 1227
 0050   8F        00      6E    00  2C 0007D            MOVC5   #0, (SP), #0, #80, (FAB)                     ; 1229
                                67     00084                                                                 :
              02  A7       34   A6  B0 00085            MOVW    52(R6), 2(FAB)                               ; 1230
                  67     5003   8F  B0 0008A            MOVW    #20483, (FAB)                                ; 1231
                                57  DD 0008F            PUSHL   FAB                                          ; 1233
       00000000G  00            01  FB 00091            CALLS   #1, SYS$CLOSE                                :
                  59            50  D0 00098            MOVL    R0, STATUS2                                  :
                  1C            59  E9 0009B            BLBC    STATUS2, 6$                                  ; 1234
                           74   A6  9F 0009E            PUSHAB  116(R6)                                      ; 1237
              04  AE       44   8F  9A 000A1            MOVZBL  #68, 4(SP)                                   :
```

**F

```
                        04      AE   9F 000A6          PUSHAB  4(SP)                              ;
                6A              02   FB 000A9          CALLS   #2, LIB$FREE_VM                    ;
                        70      A6   9F 000AC          PUSHAB  112(R6)                            ; 1238
          04    AE      50      8F   9A 000AF          MOVZBL  #80, 4(SP)                         ;
                        04      AE   9F 000B4          PUSHAB  4(SP)                              ;
                6A              02   FB 000B7          CALLS   #2, LIB$FREE_VM                    ;
                        18      A6   D5 000BA 6$:      TSTL    24(R6)                             ; 1245
                                2E   13 000BD          BEQL    7$                                 ;
                50      14      A6   D0 000BF          MOVL    20(R6), R0                         ; 1260
      51        50              08   C7 000C3          DIVL3   #8, R0, R1                         ;
          04    AE      01    A140   3E 000C7          MOVAW   1(R1)[R0], TOTAL_SPACE             ;
                        18      A6   9F 000CD          PUSHAB  24(R6)                             ; 1262
                        08      AE   9F 000D0          PUSHAB  TOTAL_SPACE                        ; 1261
                6A              02   FB 000D3          CALLS   #2, LIB$FREE_VM                    ; 1262
                58              50   D0 000D6          MOVL    R0, STATUS                         ;
                11              58   E9 000D9          BLBC    STATUS, 7$                         ; 1263
                        08      AE   9F 000DC          PUSHAB  NEXT                               ; 1269
          04    AE      78      8F   9A 000DF          MOVZBL  #120, 4(SP)                        ;
                        04      AE   9F 000E4          PUSHAB  4(SP)                              ;
                6A              02   FB 000E7          CALLS   #2, LIB$FREE_VM                    ;
                58              50   D0 000EA          MOVL    R0, STATUS                         ;
                03              59   E9 000ED 7$:      BLBC    STATUS2, 8$                        ; 1278
                58              59   D0 000F0          MOVL    STATUS2, STATUS                    ; 1280
                                6B   D4 000F3 8$:      CLRL    SCR$L_CUROUTPUT                    ; 1284
                50              58   D0 000F5          MOVL    STATUS, R0                         ; 1285
                                     04 000F8          RET                                        ; 1286

; Routine Size:  249 bytes,    Routine Base:  _LIB$CODE + 033E

; 1036          1287  1 !<BLF/PAGE>
```

```
1 %SBTTL 'CREATE - Create file via RMS'
1 ROUTINE CREATE (
1                       TCB : REF BLOCK [, BYTE],
1                       LENGTH,
1                       ADDR
1                   ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Create an output file via RMS.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = CREATE ( TCB.mab.r,
1 !                                   LENGTH.rl.r,
1 !                                   ADDR.rl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       TCB.mab.r               Current TCB address.
1 !
1 !       LENGTH.rl.r             Length of file name
1 !
1 !       ADDR.rl.r               Address of file name text string
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       Failure status from $CREATE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       The file is created and connected to.  The resulting ISI and
1 !       IFI are stored in the control block.
1 !--
1
2     BEGIN
2     LOCAL
2       FAB_BLOCK,                      ! a FAB
2       FAB : REF $FAB_DECL,            ! ptr to FAB
2       RAB_BLOCK,                      ! a RAB
2       RAB : REF $RAB_DECL,            ! ptr to RAB
2       STATUS;                         ! Status of subroutine calls
2
2 !+
2 ! Allocate the FAB and RAB here.  SCR$STOP_OUTPUT will deallocate when
2 ! the stream is stopped.
2 !-
2     STATUS = LIB$GET_VM (%REF (FAB$C_BLN), FAB_BLOCK);
```

```
	IF NOT .STATUS THEN RETURN (.STATUS);

	STATUS = LIB$GET_VM (%REF (RAB$C_BLN), RAB_BLOCK);
	IF NOT .STATUS THEN RETURN (.STATUS);

	FAB = .FAB_BLOCK;
	RAB = .RAB_BLOCK;

    !+
    ! Initialize fields in FAB prior to call to $CREATE
    !-
	CH$FILL ( 0, FAB$C_BLN, .FAB ) ;	! Clear FAB to zero
	FAB [FAB$B_BID] = FAB$C_BID ;		! Block id says its a FAB
	FAB [FAB$B_BLN] = FAB$C_BLN ;		! Length of a FAB
	FAB [FAB$B_FNS] = ..LENGTH ;		! Length of file spec
	FAB [FAB$L_FNA] = ..ADDR ;		! Address of file spec
	FAB [FAB$V_SQO] = 1 ;			! Sequential access only
	FAB [FAB$B_RFM] = FAB$C_VAR ;		! Variable-length records
	FAB [FAB$V_CR] = 1 ;			! Automatic carriage control

	IF NOT ( STATUS = $CREATE ( FAB = .FAB ))
	THEN
	    RETURN (.STATUS ) ;

    !+
    ! Initialize fields in RAB prior to $ CONNECT call.
    !-
	CH$FILL ( 0, RAB$C_BLN, .RAB ) ;	! Clear RAB to zero
	RAB [RAB$B_BID] = RAB$C_BID ;		! Block id says its a RAB
	RAB [RAB$B_BLN] = RAB$C_BLN ;		! Length of a RAB
	RAB [RAB$L_FAB] = .FAB ;		! Address of FAB

	$CONNECT ( RAB = .RAB ) ;

    !+
    ! Save IFI and ISI in caller's TCB
    !-
	TCB [SCR$W_IFI] = .FAB [FAB$W_IFI] ;
	TCB [SCR$W_ISI] = .RAB [RAB$W_ISI] ;
	TCB [SCR$L_FAB] = .FAB;
	TCB [SCR$L_RAB] = .RAB;			! save addresses for later use
	RETURN (SS$_NORMAL) ;
	END;					! End of routine CREATE


						.EXTRN	SYS$CREATE, SYS$CONNECT

			03FC 00000 CREATE:	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9		; 1289
   59 00000000G	  00	9E 00002		MOVAB	LIB$GET_VM, R9				;
   5E		  0C	C2 00009		SUBL2	#12, SP					;
		04  AE	9F 0000C		PUSHAB	FAB_BLOCK				; 1344
04 AE		50  8F	9A 0000F		MOVZBL	#80, 4(SP)				;
		04  AE	9F 00014		PUSHAB	4(SP)					;
   69		  02	FB 00017		CALLS	#2, LIB$GET_VM				;
   58		  50	D0 0001A		MOVL	R0, STATUS				;
   4B		  58	E9 0001D		BLBC	STATUS, 1$				; 1345
```

```
                                    08   AE  9F 00020          PUSHAB  RAB_BLOCK                        ; 1347
                      04   AE       44   8F  9A 00023          MOVZBL  #68, 4(SP)                       ;
                                    04   AE  9F 00028          PUSHAB  4(SP)                            ;
                           69            02  FB 0002B          CALLS   #2, LIB$GET_VM                   ;
                           58            50  D0 0002E          MOVL    R0, STATUS                       ;
                           37            58  E9 00031          BLBC    STATUS, 1$                       ; 1348
                           56       04   AE  7D 00034          MOVQ    FAB_BLOCK, FAB                   ; 1350
    0050   8F        00    6E            00  2C 00038          MOVC5   #0, (SP), #0, #80, (FAB)         ; 1356
                           66                   0003F                                                   ;
                           66     5003   8F  B0 00040          MOVW    #20483, (FAB)                    ; 1357
                      34   A6       08   BC  90 00045          MOVB    @LENGTH, 52(FAB)                 ; 1359
                      2C   A6       0C   BC  D0 0004A          MOVL    @ADDR, 44(FAB)                   ; 1360
                      04   A6       40   8F  88 0004F          BISB2   #64, 4(FAB)                      ; 1361
                      1F   A6            02  90 00054          MOVB    #2, 31(FAB)                      ; 1362
                      1E   A6            02  88 00058          BISB2   #2, 30(FAB)                      ; 1363
                                         56  DD 0005C          PUSHL   FAB                              ; 1365
                00000000G  00            01  FB 0005E          CALLS   #1, SYS$CREATE                   ;
                           58            50  D0 00065          MOVL    R0, STATUS                       ;
                           04            58  E8 00068          BLBS    STATUS, 2$                       ;
                           50            58  D0 0006B 1$:      MOVL    STATUS, R0                       ; 1367
                                         04     0006E          RET                                      ;
    0044   8F        00    6E            00  2C 0006F 2$:      MOVC5   #0, (SP), #0, #68, (RAB)         ; 1372
                           67                   00076                                                   ;
                           67     4401   8F  B0 00077          MOVW    #17409, (RAB)                    ; 1373
                      3C   A7            56  D0 0007C          MOVL    FAB, 60(RAB)                     ; 1375
                                         57  DD 00080          PUSHL   RAB                              ; 1377
                00000000G  00            01  FB 00082          CALLS   #1, SYS$CONNECT                  ;
                           50       04   AC  D0 00089          MOVL    TCB, R0                          ; 1382
                      34   A0       02   A6  B0 0008D          MOVW    2(FAB), 52(R0)                   ;
                      36   A0       02   A7  B0 00092          MOVW    2(RAB), 54(R0)                   ; 1383
                      70   A0            56  7D 00097          MOVQ    FAB, 112(R0)                     ; 1384
                           50            01  D0 0009B          MOVL    #1, R0                           ; 1386
                                         04     0009E          RET                                      ; 1387

; Routine Size:  159 bytes,    Routine Base:  _LIB$CODE + 0437

; 1138          1388  1 !<BLF/PAGE>
```

```
;  1140     1389  1 %SBTTL 'EXIT_HANDLER - Exit handler'
;  1141     1390  1 ROUTINE EXIT_HANDLER =
;  1142     1391  1 !++
;  1143     1392  1 ! FUNCTIONAL DESCRIPTION:
;  1144     1393  1 !
;  1145     1394  1 !       This routine is invoked on image exit.  It searches the list
;  1146     1395  1 !       of active streams doing a STOP_OUTPUT on each one.  Any errors
;  1147     1396  1 !       are ignored.
;  1148     1397  1 !
;  1149     1398  1 ! CALLING SEQUENCE:
;  1150     1399  1 !
;  1151     1400  1 !       ret_status.wlc.v = EXIT_HANDLER ()
;  1152     1401  1 !
;  1153     1402  1 ! FORMAL PARAMETERS:
;  1154     1403  1 !
;  1155     1404  1 !       NONE
;  1156     1405  1 !
;  1157     1406  1 ! IMPLICIT INPUTS:
;  1158     1407  1 !
;  1159     1408  1 !       SCR$L_FLINKHEAD -- the head of the list of active streams
;  1160     1409  1 !
;  1161     1410  1 ! IMPLICIT OUTPUTS:
;  1162     1411  1 !
;  1163     1412  1 !       NONE
;  1164     1413  1 !
;  1165     1414  1 ! COMPLETION STATUS:
;  1166     1415  1 !
;  1167     1416  1 !       SS$_NORMAL      Normal successful completion
;  1168     1417  1 !
;  1169     1418  1 ! SIDE EFFECTS:
;  1170     1419  1 !
;  1171     1420  1 !       NONE
;  1172     1421  1 !--
;  1173     1422  1
;  1174     1423  2     BEGIN
;  1175     1424  2
;  1176     1425  2
;  1177     1426  2     WHILE .SCR$L_FLINKHEAD NEQ 0
;  1178     1427  2     DO
;  1179     1428  3         BEGIN
;  1180     1429  3         LOCAL
;  1181     1430  3             CURRENT_TCB : REF BLOCK [, BYTE];   ! Current TCB
;  1182     1431  3
;  1183     1432  3         CURRENT_TCB = .SCR$L_FLINKHEAD ;        ! Select next TCB
;  1184     1433  3
;  1185     1434  3         SCR$SET_OUTPUT ( .CURRENT_TCB [SCR$L_STREAM]) ; ! Make current
;  1186     1435  3
;  1187     1436  3         SCR$STOP_OUTPUT () ;                            ! Stop stream
;  1188     1437  2         END ;
;  1189     1438  2     RETURN (SS$_NORMAL);
;  1190     1439  1     END;                                ! End of routine EXIT_HANDLER


                                        0000 00000 EXIT_HANDLER:
```

```
                                                   .WORD   Save nothing                          ; 1390
               50 00000000G  00  D0 00002 1$:      MOVL    SCR$L_FLINKHEAD, R0                   ; 1426
                             0F  13 00009          BEQL    2$                                    ;
                       30    A0  DD 0000B          PUSHL   48(CURRENT_TCB)                       ; 1434
          FBF5  CF           01  FB 0000E          CALLS   #1, SCR$SET_OUTPUT                    ;
          FE50  CF           00  FB 00013          CALLS   #0, SCR$STOP_OUTPUT                   ; 1436
                             E8  11 00018          BRB     1$                                    ; 1426
                50           01  D0 0001A 2$:      MOVL    #1, R0                                ; 1438
                                 04 0001D          RET                                           ; 1439

; Routine Size:  30 bytes,    Routine Base:  _LIB$CODE + 04D6

; 1191          1440  1 !<BLF/PAGE>
```

```
;  1193     1441  1 %SBTTL 'GET_CHAR - Get terminal characteristics and init TCB'
;  1194     1442  1 ROUTINE GET_CHAR (
;  1195     1443  1                     TCB : REF BLOCK [, BYTE],
;  1196     1444  1                     TYPE
;  1197     1445  1                   ) =
;  1198     1446  1 !++
;  1199     1447  1 ! FUNCTIONAL DESCRIPTION:
;  1200     1448  1 !
;  1201     1449  1 !       Get device characteristics, set up TCB and return device type.
;  1202     1450  1 !
;  1203     1451  1 ! CALLING SEQUENCE:
;  1204     1452  1 !
;  1205     1453  1 !       ret_status.wlc.v = GET_CHAR ( TCB.rab.r,
;  1206     1454  1 !                                     TYPE.wl.r )
;  1207     1455  1 !
;  1208     1456  1 ! FORMAL PARAMETERS:
;  1209     1457  1 !
;  1210     1458  1 !       TCB.rab.r               Current TCB address.
;  1211     1459  1 !
;  1212     1460  1 !       TYPE.wl.r               Returned device type.
;  1213     1461  1 !
;  1214     1462  1 !
;  1215     1463  1 ! IMPLICIT INPUTS:
;  1216     1464  1 !
;  1217     1465  1 !       NONE
;  1218     1466  1 !
;  1219     1467  1 ! IMPLICIT OUTPUTS:
;  1220     1468  1 !
;  1221     1469  1 !       NONE
;  1222     1470  1 !
;  1223     1471  1 ! COMPLETION STATUS:
;  1224     1472  1 !
;  1225     1473  1 !       SS$_NORMAL      Normal successful completion
;  1226     1474  1 !
;  1227     1475  1 ! SIDE EFFECTS:
;  1228     1476  1 !
;  1229     1477  1 !       NONE
;  1230     1478  1 !--
;  1231     1479  1
;  1232     1480  2     BEGIN
;  1233     1481  2
;  1234     1482  2 MACRO
;  1235   M 1483  2     VT52_MODE = %STRING (%CHAR(CR), %CHAR(ESC), %CHAR(LB), '?2l',
;  1236     1484  2                          %CHAR(ESC), '\', %CHAR(CR), '    ',%CHAR(CR))%,
;  1237     1485  2     VT100_MODE = %STRING (%CHAR(ESC), '<')%;
;  1238     1486  2
;  1239     1487  2     TCB [SCR$W_DEVPAGSIZ] = LIB$LP_LINES () - 6 ;
;  1240     1488  2     TCB [SCR$W_DEVWIDTH]  = 132 ;
;  1241     1489  2
;  1242     1490  2     SCR$_INFOCLASS = SCR$AB_DEVCLASS ;
;  1243     1491  2     SCR$_INFOTYPE  = SCR$AB_DEVTYPE ;
;  1244     1492  2     SCR$_INFOSIZ   = SCR$AW_DEVBUFSIZ ;
;  1245     1493  2     SCR$_INFODEP   = SCR$AL_DEVDEPEND ;
;  1246     1494  2     SCR$_INFODEP2  = SCR$AL_DEVDEPND2 ;
;  1247     1495  2
;  1248     1496  3     IF ($GETDVI ( CHAN = .TCB [SCR$W_CHAN], ITMLST = SCR$A_ITMLST ))
;  1249     1497  2     THEN
```

```
;  1250     1498  3          BEGIN    ! $GETDVI succeeded
;  1251     1499  3          TCB [SCR$B_DEVTYP] = .SCR$AB_DEVTYPE ;
;  1252     1500  3          !+
;  1253     1501  3          ! Assume BOLD and UNDERLINE supported until it proves
;  1254     1502  3          ! otherwise.
;  1255     1503  3          !-
;  1256     1504  3          TCB [SCR$L_DEVCHAR] = %X'FFFFFFF6' ;
;  1257     1505  3          IF .SCR$AB_DEVCLASS EQL DC$_TERM
;  1258     1506  3          THEN
;  1259     1507  4              BEGIN       ! Is a terminal
;  1260     1508  4              LOCAL
;  1261     1509  4                  STATUS,          ! Status of subr. calls
;  1262     1510  4                  LOC_DESC : BLOCK [8, BYTE] ;    ! Local descriptor
;  1263     1511  4
;  1264     1512  4              TCB [SCR$W_DEVWIDTH] = .SCR$AW_DEVBUFSIZ ;  ! Device width
;  1265     1513  4              TCB [SCR$W_DEVPAGSIZ] = .(SCR$AL_DEVDEPEND+3)<0,8> ; ! lines/page
;  1266     1514  4              TCB [SCR$L_DEVDEPND2] = .SCR$AL_DEVDEPND2 ;
;  1267     1515  4
;  1268     1516  4              SELECTONE .SCR$AB_DEVTYPE OF
;  1269     1517  4              SET
;  1270     1518  4                  [DT$_FT1 TO DT$_FT8]:           ! Foreign terminals
;  1271     1519  4                      .TYPE = VTFOREIGN ;
;  1272     1520  4
;  1273     1521  4                  [DT$_VT52, DT$_VT55]:           ! Treat like VT52
;  1274     1522  4                      .TYPE = VT52 ;
;  1275     1523  4
;  1276     1524  4                  [DT$_VT100]:                    ! VT100
;  1277     1525  4                      .TYPE = VT100 ;
;  1278     1526  4
;  1279     1527  4                  [DT$_VT05]:                     ! VT05
;  1280     1528  4                      .TYPE = VT05 ;
;  1281     1529  4
;  1282     1530  4                  [OTHERWISE]:                    ! Unknown
;  1283     1531  4                      IF .SCR$AL_DEVDEPND2 [TT2$V_DECCRT] OR
;  1284     1532  4                         .SCR$AL_DEVDEPND2 [TT2$V_ANSICRT]
;  1285     1533  4                      THEN
;  1286     1534  5                          BEGIN   ! VT100 compatible (ANSI)
;  1287     1535  5                          .TYPE = VT100 ;
;  1288     1536  5                          END     ! VT100 compatible (ANSI)
;  1289     1537  4                      ELSE
;  1290     1538  5                          BEGIN   ! Really Unknown
;  1291     1539  5                          .TYPE = 0 ;
;  1292     1540  5                          ! Assume NO attributes supported.
;  1293     1541  5                          TCB [SCR$L_DEVCHAR] = -1 ;
;  1294     1542  4                          END;    ! Really Unknown
;  1295     1543  4          TES;
;  1296     1544  4
;  1297     1545  4          !+
;  1298     1546  4          ! If VT52 or VT100, the terminal might be a VT100.  In any
;  1299     1547  4          ! case, issue the proper escape sequence to ensure that
;  1300     1548  4          ! the VT100 is in the correct mode, ANSI or VT52.
;  1301     1549  4          !-
;  1302     1550  4          LOC_DESC [DSC$W_LENGTH] = 0;
;  1303     1551  4          LOC_DESC [DSC$B_CLASS] = DSC$K_CLASS_S ;
;  1304     1552  4          LOC_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T ;
;  1305     1553  4          TCB [SCR$B_TYPE] = ..TYPE ;
;  1306     1554  4          IF ..TYPE EQL VT52
```

```
;  1307     1555  4              THEN
;  1308     1556  5                  BEGIN    ! To VT52 mode
;  1309     1557  5                  LOC_DESC [DSC$W_LENGTH] = %CHARCOUNT  (VT52_MODE) ;
;  1310     1558  5                  LOC_DESC [DSC$A_POINTER] = UPLIT ( BYTE  (VT52_MODE));
;  1311     1559  5                  END      ! To VT52 mode
;  1312     1560  4              ELSE
;  1313     1561  4                  IF ..TYPE EQL VT100
;  1314     1562  4                  THEN
;  1315     1563  5                      BEGIN       ! To VT100 mode
;  1316     1564  5                      LOC_DESC [DSC$W_LENGTH] = %CHARCOUNT  (VT100_MODE) ;
;  1317     1565  5                      LOC_DESC [DSC$A_POINTER] = UPLIT ( BYTE  (VT100_MODE));
;  1318     1566  4                      END;        ! To VT100 mode
;  1319     1567  4
;  1320     1568  4              IF .LOC_DESC [DSC$W_LENGTH] NEQ 0
;  1321     1569  4              THEN
;  1322     1570  5                  BEGIN
;  1323     1571  5                  STATUS = SCR$PUT_SCREEN ( LOC_DESC);
;  1324     1572  5                  IF NOT .STATUS
;  1325     1573  5                  THEN
;  1326     1574  5                      RETURN (.STATUS ) ;
;  1327     1575  4                  END;
;  1328     1576  4
;  1329     1577  4              END ! Is a terminal
;  1330     1578  3          ELSE
;  1331     1579  4              BEGIN       ! Not a terminal
;  1332     1580  4              .TYPE = 0 ; ! Mark as unknown
;  1333     1581  4              TCB [SCR$B_TYPE] = ..TYPE ;
;  1334     1582  3              END;        ! Not a terminal
;  1335     1583  3
;  1336     1584  3          END     ! $GETDVI succeeded
;  1337     1585  2      ELSE
;  1338     1586  3          BEGIN   ! $GETDVI failed
;  1339     1587  3          .TYPE = 0 ;     ! Mark unknown
;  1340     1588  3          TCB [SCR$B_TYPE] = ..TYPE ;
;  1341     1589  2          END ;   ! $GETDVI failed
;  1342     1590  2
;  1343     1591  2
;  1344     1592  2      RETURN (SS$_NORMAL) ;
;  1345     1593  1      END;                                ! End of routine GET_CHAR


                0D  20  20  20  0D  5C  1B  6C  32  3F  5B  1B  0D  004F4 P.AAB:  .ASCII  <13><27>\[?2l\<27><92><13>\   \<13>          ;
                                                            00501        .BLKB   3
                                                        3C  1B  00504 P.AAC:  .ASCII  <27>\<\                                       ;

                                                                             .EXTRN  SYS$GETDVI

                                                            001C 00000 GET_CHAR:
                                                                             .WORD   Save R2,R3,R4                                      ; 1442
                                        54 00000000'        EF   9E 00002        MOVAB   SCR$AL_DEVDEPND2, R4                           ;
                                        5E                  08   C2 00009        SUBL2   #8, SP                                         ;
                                        52          04      AC   D0 0000C        MOVL    TCB, R2                                        ; 1487
                    00000000G           00              FB  00010        CALLS   #0, LIB$LP_LINES                               ;
                0E  A2                  50              06  A3  00017        SUBW3   #6, R0, 14(R2)                                 ;
                                    0C  A2          84  8F  9B  0001C        MOVZBW  #132, 12(R2)                                   ; 1488
                                    08  A4          F0  A4  9E  00021        MOVAB   SCR$AB_DEVCLASS, SCR$_INFOCLASS                ; 1490
```

```
              14    A4         F4   A4  9E 00026        MOVAB   SCR$AB_DEVTYPE, SCR$_INFOTYPE               ; 1491
              20    A4         F8   A4  9E 0002B        MOVAB   SCR$AW_DEVBUFSIZ, SCR$_INFOSIZ              ; 1492
              2C    A4         FC   A4  9E 00030        MOVAB   SCR$AL_DEVDEPEND, SCR$_INFODEP              ; 1493
              38    A4              64  9E 00035        MOVAB   SCR$AL_DEVDEPND2, SCR$_INFODEP2             ; 1494
                    53         08   AC  D0 00039        MOVL    TYPE, R3                                    ; 1553
                                    7E  7C 0003D        CLRQ    -(SP)                                       ; 1496
                                    7E  7C 0003F        CLRQ    -(SP)                                       ;
                               04   A4  9F 00041        PUSHAB  SCR$A_ITMLST                                ;
                                    7E  D4 00044        CLRL    -(SP)                                       ;
                    7E         08   A2  3C 00046        MOVZWL  8(R2), -(SP)                                ;
                                    7E  D4 0004A        CLRL    -(SP)                                       ;
         00000000G  00              08  FB 0004C        CALLS   #8, SYS$GETDVI                              ;
                    03              50  E8 00053        BLBS    R0, 2$                                      ;
                             00AC   31 00056 1$:        BRW     11$                                         ;
                    50         F4   A4  D0 00059 2$:    MOVL    SCR$AB_DEVTYPE, R0                          ; 1499
              0B    A2              50  90 0005D        MOVB    R0, 11(R2)                                  ;
              10    A2              0A  CE 00061        MNEGL   #10, 16(R2)                                 ; 1504
       00000042     8F         F0   A4  D1 00065        CMPL    SCR$AB_DEVCLASS, #66                        ; 1505
                                    E7  12 0006D        BNEQ    1$                                          ;
              0C    A2         F8   A4  B0 0006F        MOVW    SCR$AW_DEVBUFSIZ, 12(R2)                    ; 1512
              0E    A2         FF   A4  9B 00074        MOVZBW  SCR$AL_DEVDEPEND+3, 14(R2)                  ; 1513
              44    A2              64  D0 00079        MOVL    SCR$AL_DEVDEPND2, 68(R2)                    ; 1514
                    10              50  D1 0007D        CMPL    R0, #16                                     ; 1518
                                    0B  19 00080        BLSS    3$                                          ;
                    17              50  D1 00082        CMPL    R0, #23                                     ;
                                    06  14 00085        BGTR    3$                                          ;
              08    BC              04  D0 00087        MOVL    #4, @TYPE                                   ; 1519
                                    3E  11 0008B        BRB     8$                                          ;
                    3F              50  D1 0008D 3$:    CMPL    R0, #63                                     ; 1521
                                    0F  15 00090        BLEQ    4$                                          ;
       00000041     8F              50  D1 00092        CMPL    R0, #65                                     ;
                                    06  14 00099        BGTR    4$                                          ;
              08    BC              02  D0 0009B        MOVL    #2, @TYPE                                   ; 1522
                                    2A  11 0009F        BRB     8$                                          ;
       00000060     8F              50  D1 000A1 4$:    CMPL    R0, #96                                     ; 1524
                                    14  13 000A8        BEQL    6$                                          ;
                    01              50  D1 000AA        CMPL    R0, #1                                      ; 1527
                                    06  12 000AD        BNEQ    5$                                          ;
              08    BC              01  D0 000AF        MOVL    #1, @TYPE                                   ; 1528
                                    16  11 000B3        BRB     8$                                          ;
   04         03    A4              05  E0 000B5 5$:    BBS     #5, SCR$AL_DEVDEPND2+3, 6$                  ; 1531
                    06         03   A4  E9 000BA        BLBC    SCR$AL_DEVDEPND2+3, 7$                      ; 1532
              08    BC              03  D0 000BE 6$:    MOVL    #3, @TYPE                                   ; 1535
                                    07  11 000C2        BRB     8$                                          ; 1531
                               08   BC  D4 000C4 7$:    CLRL    @TYPE                                       ; 1539
              10    A2              01  CE 000C7        MNEGL   #1, 16(R2)                                  ; 1541
                    6E  010E0000    8F  D0 000CB 8$:    MOVL    #17694720, LOC_DESC                         ; 1550
              0A    A2              63  90 000D2        MOVB    (R3), 10(R2)                                ; 1553
                    02              63  D1 000D6        CMPL    (R3), #2                                    ; 1554
                                    0B  12 000D9        BNEQ    9$                                          ;
                    6E              0D  B0 000DB        MOVW    #13, LOC_DESC                               ; 1557
              04    AE       FF0C   CF  9E 000DE        MOVAB   P.AAB, LOC_DESC+4                           ; 1558
                                    0E  11 000E4        BRB     10$                                         ; 1554
                    03              63  D1 000E6 9$:    CMPL    (R3), #3                                    ; 1561
                                    09  12 000E9        BNEQ    10$                                         ;
                    6E              02  B0 000EB        MOVW    #2, LOC_DESC                                ; 1564
              04    AE       FF0C   CF  9E 000EE        MOVAB   P.AAC, LOC_DESC+4                           ; 1565
```

```
                                    6E  B5 000F4 10$:    TSTW    LOC_DESC                                ; 1568
                                    13  13 000F6          BEQL    12$                                     ;
                                    5E  DD 000F8          PUSHL   SP                                      ; 1571
                         00000000G  00  01 000FA          CALLS   #1, SCR$PUT_SCREEN                      ;
                                    07  50 E8 00101       BLBS    STATUS, 12$                             ; 1572
                                        04 00104          RET                                             ; 1574
                                    63  D4 00105 11$:     CLRL    (R3)                                    ; 1587
                                 0A A2  63 90 00107       MOVB    (R3), 10(R2)                            ; 1588
                                    50  01 D0 0010B 12$:  MOVL    #1, R0                                  ; 1592
                                        04 0010E          RET                                             ; 1593

; Routine Size:  271 bytes,    Routine Base:  _LIB$CODE + 0506

; 1346          1594  1 !<BLF/PAGE>
```

```
; 1348        1595  1 END                                        ! End of module LIB$SCREEN
; 1349        1596  1
; 1350        1597  0 ELUDOM




                                                        LIB$STOP_OUTPUT==    SCR$STOP_OUTPUT

;                                   PSECT SUMMARY
;
;      Name                     Bytes                          Attributes
;
;   _LIB$DATA                       104  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   _LIB$CODE                      1557  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)




;                            Library Statistics
;
;                                               -------- Symbols --------      Pages       Processing
;      File                                     Total    Loaded   Percent      Mapped      Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1            9776        65         0        581        00:01.0
;   _$255$DUA28:[VMSLIB.OBJ]SCRLIB.L32;1           62        35        56          7        00:00.1




;                                COMMAND QUALIFIERS

;      BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:SCRMISC/OBJ=OBJ$:SCRMISC MSRC$:SCRMISC/UPDATE=(ENH$:SCRMISC)

; Size:           1528 code + 133 data bytes
; Run Time:          00:31.8
; Elapsed Time:      00:33.8
; Lines/CPU Min:     3013
; Lexemes/CPU-Min: 20915
; Memory Used:  229 pages
; Compilation Complete
```

SYSGMSG
LIS

SCRINPUT
LIS

SCRMISC
LIS

SCRREQ
LIS

SETPRIV
LIS

SHRMSG
LIS

TPARSE
LIS

SCRTIB
LIS

SCRVECTOR
LIS

SSMSG
LIS